A Cooperative Approach for Composite Ontology Matching

Cássia Trojahn dos Santos

Advisor: Prof. Doutor Paulo Quaresma

Co-Advisor: Profa. Doutora Renata Vieira

Thesis submitted to University of Évora
to obtention of the degree of PhD in Informatics

Department of Informatics
University of Évora

November, 2008

This version does not have the comments of the jury

A Cooperative Approach for Composite Ontology Matching

Cássia Trojahn dos Santos

Advisor: Prof. Doutor Paulo Quaresma

Co-Advisor: Profa. Doutora Renata Vieira

Department of Informatics
University of Évora

November, 2008

This version does not have the comments of the jury

Departamento de Informática
Universidade de Évora

Tese de Doutoramento:

Uma Abordagem Cooperativa para Mapeamento de Ontologias

Cássia Trojahn dos Santos

Resumo Alargado

Prof. Doutor Paulo Quaresma - Universidade de Évora, Portugal
(Orientador)

Profa. Doutora Renata Vieira - Pontifícia Universidade Católica do Rio Grande do Sul, Brasil
(Co-orientadora)

Évora, Novembro de 2008.

1 Introdução

Resolver o problema de heterogeneidade semântica é crucial para permitir interoperabilidade entre sistemas baseados em ontologias. Isto torna o processo automático de mapeamento de ontologias, como uma antecipada solução para a heterogeneidade semântica, uma importante área de pesquisa.

Diferentes abordagens para o problema de mapeamento têm sido propostas na literatura. A principal diferença entre essas técnicas corresponde à maneira com as quais elas exploram as características das ontologias. Enquanto abordagens sintácticas consideram medidas de similaridade entre cadeias de caracteres, abordagens semânticas utilizam relações semânticas, usualmente com base em recursos linguísticos. Outras abordagens consideram as posições dos termos na hierarquia da ontologia ou instâncias das mesmas ([8]). Entretanto, cada categoria oferece uma variedade de opções. Tais abordagens têm sido apresentadas sob diferentes perspectivas em ([21]), ([30]), ([12]), ([22]), ([23]), ([2]), and ([5]).

Os sistemas de mapeamento automático pressupõe que o uso de técnicas simples de mapeamento não são suficientes para o problema e diversas técnicas são agregadas em um processo unificado. Tal agregação envolve execução sequencial ou paralela de diversos mapeadores, cujos resultados variam de uma simples soma ponderada á aprendizagem do melhor mapeador. Além disso, algumas técnicas executam melhor do que outras em casos específicos, dependendo de como as técnicas fazem uso dos recursos disponíveis, bem como abordagens com bom desempenho em um caso específico podem não ter sucesso em outros. Um problema importante em mapeamento de ontologias consiste em encontrar formas efectivas de escolha entre as diversas técnicas e suas variações, e então combinar os seus resultados.

Uma promissora opção consiste em formalizar a combinação de técnicas de mapeamento usando abordagens baseadas em agentes inteligentes, tais como negociação cooperativa e argumentação. Seguindo tal perspectiva, diferentes agentes utilizam abordagens particulares, obtendo resultados distintos que devem ser compartilhados, comparados, escolhidos e aceites. Tais abordagens permitem combinar abordagens de mapeamento de forma efectiva. Primeiro, os resultados finais da negociação e argumentação representam um consenso entre as diferentes visões. Segundo, resolvendo conflitos os resultados individuais podem ser melhorados. Terceiro, agentes podem ser agregados e substituídos de modo a representar abordagens mais adequadas para o caso específico, permitindo ajustar a configuração da solução.

Nesta tese, a formalização do problema de mapeamento automático de ontologias, seguindo uma abordagem baseada em agentes, nomeadamente negociação cooperativa e argumentação é apresentada e avaliada. A avaliação de ambos os modelos é feita a partir da utilização de estudos de casos aceites pela comunidade científica. Uma comparação entre negociação e argumentação é feita, bem como os seus resultados são avaliados contra sistemas de mapeamento de ontologias actuais.

2 Abordagens Cooperativas para Mapeamento de Ontologias

Pesquisas em Sistemas Multiagentes (SMAs) são motivadas pela contextualização, projecto, e implementação de entidades (agentes) que operam em um ambiente aberto e distribuído. De acordo com Sycara ([25]), os SMAs são caracterizados pelos seguintes aspectos: (i) cada agente tem informações ou capacidades de resolver o problema incompletas, e então, um ponto de vista limitado; (ii) não há controle global no sistema; (iii) informações são descentralizadas; e (iv) computação é assíncrona. De forma a resolver os conflitos que podem derivar da cooperação e coordenação entre agentes, mecanismos de negociação e argumentação são utilizados. Ambas abordagens envolvem algum tipo de comunicação entre agentes. Entretanto, sistemas envolvendo negociação e argumentação são diferentes em sua natureza, variando a comunicação em forma de leilões á argumentação no senso mais filosófico.

Negociação envolve comunicação entre agentes que iterativamente trocam propostas e contra-propostas. Argumentação pode ser vista com uma forma mais sofisticada de troca de propostas em um protocolo de negociação ou em um modelo de raciocínio baseado na construção e comparação de argumentos. Sob a primeira perspectiva, argumentos são vistos como meta-informações usadas para justificar as propostas ou persuadir agentes a trocar suas propostas. Na segunda perspectiva, argumentação pode ser abstractamente definida como a interacção de diferentes argumentos em favor ou contra conclusões.

Nesta tese, ambas negociação e argumentação propostas são aplicadas ao problema de mapeamento automático de ontologias. Usando negociação, os agentes interagem seguindo um protocolo baseado em votos, onde cada proposta representa um voto contra ou a favor de um mapeamento entre termos das ontologias. Deste modo, o consensus é baseado em votos. Usando argumentação, mapeamentos são representados como argumentos. De acordo com as relações de ataques, um argumento em favor de um mapeamento, gerado por um mapeador, pode ser suportado ou atacado por argumentos gerados por outros mapeadores. Baseado na instanciação de modelos de argumentação (usando relações de ataque específicas e ordem de preferência entre argumentos), os mapeadores calculam seus conjuntos de argumentos preferíveis. Os argumentos em tais conjuntos são vistos como o conjunto de argumentos (mapeamentos) globalmente aceites. A seguir, ambos os modelos são detalhados.

2.1 Negociação Cooperativa

No modelo de negociação ([29], [28]), os agentes encapsulam diferentes abordagens de mapeamento de ontologias to mapear duas ontologias diferentes. Os resultados individuais são então compartilhados, comparados, e aceites, e um alinhamento final é obtido.

2.1.1 Processo de Negociação

Basicamente, o processo de negociação envolve duas fases. Primeiramente, os agentes trabalham de forma independente, aplicando abordagens específicas de mapeamento e gerando conjuntos de objectos de negociação. Um objecto de negociação é um 2-tuplo (A, m), onde A indica o agente mapeador gerando o mapeamento m. Um mapeamento m é um 5-tuplo (e,e',h,R,s), onde

e e *e'* são entidades das ontologias; *h* é um valor +,- indicando se o mapeamento entre *e* e *e'* é obtido ou não, respectivamente; *R* é a relação entre *e* e *e'*; e *s* é um valor contínuo indicando o grau de confiança no mapeamento, o qual pode ser usado para filtrar mapeamentos com grau de confiança abaixo de um limiar. Segundo, o conjunto de objectos de negociação é negociado entre os agentes. O processo de negociação envolve um agente mediador e agentes mapeadores.

Considera-se, por exemplo, a interacção entre agentes para o mapeamento das seguintes entidades, *e* = "personal computer" and *e'* = "pc", onde três mapeadores são usados, nomeadamente sintáctico, semântico, e estrutural. Esses mapeadores geram os seguintes mapeamentos, respectivamente: $m_{sintctico}$ = ("personal-computer","pc",-,exactMatch,1.0), $m_{semntico}$ = ("personal-computer","pc",+,exactMatch,1.0), and $m_{estructural}$ = ("personal- computer", "pc",+,exactMatch,1.0).

O processo de negociação inicia com o agente mediador solicitando aos agentes mapeadores o número de seus mapeamentos positivos. O primeiro agente a gerar uma proposta é aquele que possui o maior número de mapeamentos positivos (considere, por exemplo, que o agente sintáctico tem o maior número de mapeamentos).

A proposta contem o primeiro objecto de negociação que ainda não foi avaliado pelo agente. Esta proposta é então enviada ao mediador, o qual envia a mesma para os agentes mapeadores (no caso específico o agente sintáctico propõe um mapeamento negativo, *h* = -, para o mapeamento entre as entidades "personal computer" e "pc"). Cada agente então avalia a proposta, seleccionando um objecto de negociação equivalente. Um objecto de negociação é equivalente a outro se ambos referem-se as mesmas entidades *e* e *e'* nas duas ontologias.

Se o objecto de negociação equivalente tem o mesmo valor de *h*, o agente aceita a proposta. Caso contrário, o objecto de negociação é enviado ao mediador, como uma contra-proposta. O mediador então avalia as diversas possíveis contra-propostas aceitas. No exemplo, os agentes semântico e estrutural geram contra-propostas, indicando mapeamentos positivos (*h* = +) entre as entidades sendo comparadas. Enquanto o agente semântico identifica que as entidades são sinonimas no WordNet, o agente estrutural identifica que as entidades têm super-classes similares.

O mediador selecciona a contra-proposta com maior número de votos. Se mapeamentos contraditórios (positivo e negativo) recebem o mesmo número de votos, o mediador usa uma ordem de preferência global (por exemplo, semântico > estrutural > sintáctico) para seleccionar o resultado final. Quando uma proposta é aceite por todos os agentes ou um consensus em uma contra-proposta é obtido, o mediador adiciona o objecto de negociação correspondente em um conjunto de consenso de negociação e os agentes mapeadores marcam os objectos equivalentes como avaliados. O processo de negociação encerra quando todos os objectos de negociação têm sido avaliados.

Além disso, quando os mapeadores indicam relações de mapeamento diferentes para as entidades correspondentes, a relação indicada pelo mapeador com maior preferência é seleccionada.

2.2 Modelos de Argumentação

No modelo de argumentação proposto, argumentos em favor ou contra mapeamentos entre entidades de ontologias são representados e processados. Mapeadores encapsulando diferentes

categorias de abordagens de mapeamento, geram um conjunto de argumentos. De acordo com as definições de ataque, um argumento para um mapeamento, gerado por um mapeador, pode ser suportado ou atacado por argumentos gerados por outros mapeadores. De acordo com as relações de ataques, um argumento em favor de um mapeamento, gerado por um mapeador, pode ser suportado ou atacado por argumentos gerados por outros mapeadores. Baseado na instanciação de modelos de argumentação (usando relações de ataque específicas e ordem de preferência entre argumentos), os mapeadores calculam seus conjuntos de argumentos preferíveis. Os argumentos em tais conjuntos são vistos como o conjunto de argumentos (mapeamentos) globalmente aceites.

O modelo de argumentação proposto no contexto desta tese – *Strength-based Argumentation Framework* (S-VAF) – é baseado no *Value-based Argumentation Framework* (VAF) ([1]). O VAF é estendido de forma a representar argumentos com graus. Originalmente, o VAF permite determinar quais argumentos são aceitáveis, considerando-se diferentes *audiências*, as quais podem caracterizar diferentes preferências entre os agentes. Entretanto, a noção de aceitabilidade de argumentos é baseada na preferência entre audiência. A qualidade de cada argumento isoladamente não é considerada.

De forma a definir a noção de aceitabilidade baseada na qualidade do argumento e no contexto em que ele ocorre, o VAF é estendido com graus de confiança que representam a qualidade do argumento. Uma audiência representa a ordem de preferência entre os diferentes mapeadores (por exemplo, o agente semântico é preferido em relação ao agente sintáctico). A ideia do S-VAF é obter um conjunto de argumentos que são aceitáveis por todas as audiências, considerando a qualidade do argumento. A seguir, o S-VAF é detalhado.

2.3 Modelo de Argumentação baseado em Graus de Confiança

Os graus de confiança representam a confiança que um agente tem no argumento correspondente. Um elemento tem sido adicionado ao VAF: uma função que mapeia argumentos a valores reais no intervalo [0,1]. Tal medida é um critério relevante no domínio de mapeamento de ontologias. Muitas ferramentas de mapeamento retornam mapeamentos com valores representando a confiança que elas possuem na similaridade das entidades sendo mapeadas. Tal confiança é usualmente derivada de avaliações de similaridade realizadas durante o processo de mapeamento, por exemplo, a partir da distância de edição entre labels, ou sobreposição entre conjuntos de instâncias. Um S-VAF tem as seguintes definições:

Definição 1 Um S-VAF é um 6-tuplo *(AR,attacks,V,val,P,valS)* onde *(AR,attacks,V,val,P)* é um VAF, sendo composto por um conjunto de argumentos (*AR*), um conjunto de relações de ataque (*attacks*), *V* é um conjunto não vazio de valores, *val* é uma função que mapeia elementos de *AR* aos elementos de *V*, e *P* é um conjunto de possível audiências. *valS* é uma função que mapeia elementos de *AR* a valores reais no intervalo [0,1], representando a confiança do argumento.

Definição 2 Um argumento $x \in AR$ ataca$_a$ um argumento $y \in AR$ para a audiência a se e somente se $attacks(x,y) \wedge ((valS(x) > valS(y)) \vee (\neg\, valpref(val(y),val(x)) \wedge (\neg\, (valS(y) > valS(x)))))$.

Um ataque é um ataque com sucesso se (a) a confiança do argumento atacador é maior que a confiança do argumento sendo atacado; ou se (b) o argumento sendo atacado não possui maior preferência que o argumento atacador (ou se ambos os argumentos referem-se ao mesmo valor de preferência) e a confiança do argumento sendo atacado não é maior que a preferência do argumento atacador.

Definição 3 Um argumento $A \in AR$ é *aceitável* para a audiência a ($aceitável_a$) com respeito ao conjunto de argumentos S, $aceitável_a(A,S)$ se $(\forall x)((x \in AR \ \& \ ataca_a(x,A)) \longrightarrow (\exists y)((y \in S) \ \& \ ataca_a(y,x)))$.

Definição 4 Um conjunto S de argumentos é *livre de conflito* para a audiência a se $(\forall x)(\forall y)((x \in S \wedge y \in S) \longrightarrow (\neg attacks(x, y) \vee (\neg(valS(x) > valS(y)) \wedge (valpref(val(y), val(x)) (\vee (valS(y) > valS(x))))))$.

Definição 5 Um conjunto de argumentos S no S-VAF é uma *extensão preferida* para a audiência a ($preferida_a$) se ele é o máximo (com relação ao conjunto de inclusão) *admissível* para a audiência a de AR.

Definição 6 Um argumento $x \in AR$ é *subjectivamente* aceitável se e somente se x aparece na extensão preferida de algumas audiências, mas não em todas. Um argumento $x \in AR$ é *objectivamente* aceitável se e somente se x aparece na extensão preferida de cada audiência. Um argumento que não é subjectivamente ou objectivamente aceitável é considerado *indefensível*.

Se argumentos objectivamente aceitáveis referem-se ao mesmo tipo de mapeamento (positivo ou negativo), tais argumentos são considerados como um *consenso global*, i.e., os argumentos que têm o mesmo valor de h em todas as extensões preferidas. Por outro lado, *consenso local* refere-se ao conjunto de mapeamentos que estão em alguma extensão preferida. Isto leva às definições de *consenso global* e *consenso local*:

Definição 7 Um argumento $x \in AR$ está no *consenso global* se e somente se x aparece na extensão preferida de cada audiência ou se o mapeamento $m \in x$ tem o mesmo valor de h em cada extensão preferida. Um argumento $x \in AR$ está no *consenso local* se e somente se x aparece na extensão preferida de alguma audiência ou se o mapeamento $m \in x$ tem o mesmo valor de h em alguma extensão preferível. Um argumento que não está em algum destes conjuntos, é dito *indefensível*.

2.3.1 Processo de Argumentação

Em um S-VAF, os valores $v \in V$ representam diferentes abordagens usadas pelos agentes. Três abordagens são consideradas: sintáctica (L), semântica (S), e estrutural (E), de forma que $V = \{L,S,E\}$. Cada audiência tem uma ordem de preferência entre os valores. Por exemplo, o agente sintáctico representa uma audiência onde os valor L é preferível em relação aos valores S e E. A ideia é não haver uma audiência individual com preferência entre os agentes (i.e., agentes

semânticos são preferidos em relação a todos os outros agentes), mas tentar acomodar diferentes audiências e suas preferências.

O processo de argumentação tem duas fases principais: *geração de argumentos* e *geração da extensão preferível.* Primeiramente, os agentes trabalham de forma independente, aplicado abordagens de mapeamento específicas e gerando os conjuntos de alinhamento. Um alinhamento consiste de um conjunto de todas as possíveis correspondências (mapeamentos) entre entidades de duas ontologias. Um mapeamento m é descrito como um 5-tuplo $m = (e,e',h,R,s)$, onde e corresponde a uma entidade da ontologia 1, e' corresponde a uma entidade da ontologia 2, h é um dos valores {-,+}, dependendo do tipo de mapeamento, R é a relação de mapeamento resultando do mapeamento entre as duas entidades, e s é a *confiança* associada ao mapeamento. Cada mapeamento m é encapsulado em um argumento *arg*. Um argumento $arg \in AR$ é um 2-tuplo $x = (m,a)$, onde m é um mapeamento; $a \in V$ é o valor associado ao argumento, dependendo do agente gerando esse argumento.

A confiança de um argumento é definida pelo agente quando aplicando a técnica de mapeamento específica. Após a geração de seus conjuntos de argumentos, os agentes trocam com cada outro tais conjuntos. Quando todos os agentes têm recebido os conjuntos de argumentos de cada outro, eles geram as suas relações de ataque. Um ataque (ou contra-argumento) ocorre quando argumentos para um mapeamento envolvendo as mesmas entidades têm valores conflitantes de h. Por exemplo, um argumento $x = (m_1,L)$, onde m_1 = (e,e',+,1.0,*equivalence*), tem como um ataque um argumento $y = (m_2,E)$, onde m_2 = (e,e',-,1.0,*equivalence*). m_1 e m_2 refere-se as mesmas entidades e e e' nas ontologias. O argumento y também representa um ataque ao argumento x.

Quando os conjuntos de argumentos e ataques têm sido produzidos, os agentes definem quais argumentos devem ser aceites, considerando a audiência específica. Para isso, os agentes computam suas extensões preferíveis, de acordo com as audiências e confiança dos argumentos. Baseado nas extensões geradas, os conjuntos de consensos locais e globais são definidos.

Finalmente, quando dois mapeadores indicam diferentes relações de mapeamento para as entidades correspondentes, a relação indicada pelo mapeador com maior preferência é seleccionada.

3 Experimentos

De forma a avaliar o desempenho dos modelos propostos, dois conjuntos de dados são utilizados: (i) um *benchmark* de ontologias de domínio bibliográfico, provido pelo *Ontology Alignment Evaluation Initiative* (OAEI)[1]; e (ii) um estudo de caso real, provido pelo *National Library of the Netherlands*. Para o *benchmark*, duas configurações de avaliação são especificadas, dependendo do tipo de entrada usada pelos modelos de negociação e argumentação: (i) resultados de três mapeadores propostos, nomeadamente, sintáctico, semântico, e estrutural; e (ii) resultados de sistemas participantes da campanha de avaliação OAEI 2007 *benchmark*.

[1]http://oaei.ontologymatching.org/

3.1 Caso 1: Benchmark e Mapeadores Propostos

Os mapeadores propostos são baseados, basicamente, nas seguintes abordagens: distância de edição ([16]) entre cadeias de caracteres (sintáctico), uso da base de dados WordNet[2] para consultar entidades sinónimas, e comparação de super-classes das entidades sendo mapeadas. Um mapeador baseado na análise dos tipos de dados usados para descrever os atributos é usado, especificamente, para o caso de mapeamento de atributos das ontologias.

Nesta secção, inicialmente, diferentes valores de confiança para argumentos que representam mapeamentos negativos (h = -) são considerados para verificar o comportamento do modelo de argumentação. Após, os modelos de negociação e argumentação são analisados com relação ao *baseline* – o qual é composto pela união de todos os mapeamentos – e resultados individuais dos mapeadores.

No modelo de negociação, quando mapeamentos positivos e negativos recebem o mesmo número de votos, o voto do agente sintáctico é usado para resolver o conflito. O resultado da argumentação contem somente os argumentos objectivamente aceitáveis e as audiências representam as seguintes ordens completas, definidas de acordo com os desempenhos individuais dos mapeadores: audiência *sintáctica* – sintáctica > semântica > estrutural; audiência *semântica* – semântica > sintáctica > estrutural; audiência *estrutural* – estrutural > sintáctica > semântica; audiência *atributos* – atributos > sintáctica > semântica.

Primeiramente, dois valores são usados para representar a confiança de contra-argumentos de um mapeamento positivo: 0.5 e 1.0. A Tabela 1 apresenta os resultados. Considerando que os mapeadores geram argumentos para mapeamentos positivos com confiança entre 0.80 e 1.0, o uso da confiança 0.5 para mapeamentos negativos não representa ataques aos mapeamentos positivos. Quando os mapeadores têm bom desempenho, isto resulta melhores valores de abrangência, pois todos os mapeamentos são seleccionados (caso das classes (C) da Tabela 1, onde significante diferença é observada entre os valores de F–measures^{macro} para 0.5 e 1.0). Por outro lado, se os mapeadores possuem desempenho insuficiente, os mapeamentos negativos dos bons mapeadores não representam ataques aos mapeamentos falsos negativos, resultando uma baixa precisão (caso dos atributos (A), onde os mapeamentos verdadeiros negativos dos mapeadores sintáctico e semântico não atacam os mapeamentos falsos positivos do mapeador de atributos).

Quando usando valores de 1.0 para os mapeamentos negativos, os falsos positivos mapeamentos gerados pelos mapeadores com baixo desempenho são atacados ou não seleccionados como objectivamente aceitáveis. Neste caso, a precisão é significantemente melhor, enquanto a abrangência representa a mais baixa abrangência relativa aos melhores mapeadores. Este é o caso do mapeador estrutural (coluna C), o qual tem argumentos que atacam com sucesso os argumentos dos agentes sintáctico e semântico. Para a configuração (All) (classes + atributos), ambos F–measure^{macro} e F–measure^{micro} para confiança igual 1.0 para os mapeamentos negativos são melhores do que os valores de F–measures quando usando 0.5.

Após, os resultados considerando confiança de 1.0 para mapeamentos negativos são usados. A Tabela 2 apresenta os resultados para o *baseline*, negociação e argumentação. Conforme o esperado, *baseline* tem maior abrangência que os modelos de negociação e argumentação. Por

[2]http://wordnet.princeton.edu/

Tabela 1: Diferentes valores de confiança para os mapeamentos negativos – mapeadores propostos.

	0.5			1		
	C	A	All	C	A	All
P^{macro}	**1.0**	0.02	0.15	**1.0**	**0.80**	**0.90**
P^{micro}	**1.0**	0.03	0.04	**1.0**	**0.99**	**0.99**
R^{macro}	**0.54**	**0.92**	**0.70**	0.33	0.49	0.39
R^{micro}	**0.53**	**0.95**	**0.77**	0.33	0.49	0.42
F^{macro}	**0.86**	0.05	0.21	0.63	**0.55**	**0.54**
F^{micro}	**0.69**	0.05	0.08	0.50	**0.66**	**0.59**

Tabela 2: Resultados para *baseline*, negociação, e argumentação (confiança 1.0) – mapeadores propostos.

	Baseline			Negociação			Argumentação		
	C	A	All	C	A	All	C	A	All
P^{macro}	**0.99**	0.02	0.15	**1.0**	**0.71**	**0.84**	**1.0**	**0.80**	**0.90**
P^{micro}	**1.0**	0.03	0.04	**1.0**	**0.89**	**0.93**	**1.0**	**0.99**	**0.99**
R^{macro}	**0.54**	**0.92**	**0.70**	**0.52**	0.54	0.52	0.33	0.49	0.39
R^{micro}	**0.53**	**0.95**	**0.77**	**0.52**	0.55	0.54	0.33	0.49	0.42
F^{macro}	**0.86**	0.05	0.21	**0.84**	**0.56**	**0.65**	0.63	**0.55**	**0.54**
F^{micro}	**0.69**	0.05	0.08	**0.68**	**0.68**	**0.68**	**0.50**	**0.66**	**0.59**

outro lado, *baseline* produz baixos valores de precisão, especialmente na presença de vários mapeamentos falsos positivos – casos onde o mapeador atributo é usado, colunas (A) e (All) das tabelas. Nestes casos, a precisão da negociação é significantemente melhor do que a precisão do *baseline*. Para o caso mais interessante, (All), melhores valores de F–measure são obtidos pelos modelos de negociação e argumentação, quando comparados ao *baseline*.

Comparando os modelos de negociação e argumentação, não existem diferenças significativas entre os valores de precisão e abrangência, e consequentemente F–measure (excepção é a F–measure^{macro} para as classes (C), onde negociação tem melhor desempenho). Entretanto, um mapeamento falso positivo que não é aceitável por um mapeador, pode ser aceito por voto no processo de negociação, o que pode não ocorre no processo de argumentação (o contra-argumento correspondente não é aceitável em todas as audiências). Deste modo, o processo de argumentação pode filtrar alguns mapeamentos falsos positivos (melhorando a precisão) mas, por outro lado, pode eliminar alguns mapeamentos verdadeiros positivos aceites apenas por um mapeador, reduzindo a abrangência.

Analisando os resultados dos modelos de negociação e argumentação com os resultados individuais dos mapeadores, quando considerando somente o mapeamento de classes, os mapeadores produzem conjuntos de mapeamentos similares, e o resultado dos modelos de negociação e argumentação é similar aos resultados individuais. Para o mapeamento dos atributos, os modelos propostos produzem resultados significantemente melhores, quando comparados com o mapeador atributo. Este mapeador produz um grande número de mapeamentos falsos positivos, os quais não são produzidos pelos demais mapeadores. Deste modo, usando negociação e argumentação

parte destes mapeamentos são filtrados.

3.2 Caso 2: Benchmark e Mapeadores da Avaliação OAEI

Nesta configuração, os seguintes sistemas são considerados[3]: ASMOV ([9], [10]), DSSim ([18]), Falcon ([7], [11], [20]), Lily ([31]), Ola ([6]), OntoDNA ([13], [14]), PriorPlus ([17]), RiMON ([27]), Sambo ([15], [26]), SEMA ([24]), TaxoMap ([32]), e XSOM ([3], [4]). DSSim, OntoDNA, PriorPlus, TaxoMap, e XSOM são baseados no uso de informações tais como labels das classes, propriedades e hierarquia da ontologia, enquanto os sistemas ASMOV, Falcon, Lily, Ola, RiMON, Sambo, e SEMA usam tais informações, juntamente com instâncias das ontologias.

Usando o modelo de negociação, quando mapeamentos positivos e negativos recebem o mesmo número de votos, o voto do sistema ASMOV (melhor desempenho individual) é usado para resolver o impasse. No modelo de argumentação, somente os argumentos objectivamente aceitáveis, e as audiências representam a seguinte ordem completa (exemplo): audiência *ASMOV* – ASMOV > Lily > RiMON > Falcon > Ola > PriorPlus > Sema > DSSim > XSom > Sambo > OntoDNA; audiência *Lily* – Lily > ASMOV > RiMON > Falcon > Ola > PriorPlus > Sema > DSSim > XSom > Sambo > OntoDNA; e assim por diante.

Dois valores são usados para representar a confiança de contra-argumentos de mapeamentos positivos: 0.5 e 1.0. A Tabela 3 apresenta os resultados. Basicamente, os mapeadores produzem mapeamentos positivos com confiança entre 0.80 e 1.0. Considerando que os mapeadores têm bom desempenho, isto produz melhores valores de abrangência (a maioria dos mapeamentos verdadeiros positivos são seleccionados). Entretanto, alguns mapeamentos falsos positivos são seleccionados porque os mapeamentos verdadeiros positivos não representam ataques, resultando baixos valores de precisão.

Quando usando um valor de 1.0, os mapeamentos falsos positivos dos mapeadores com baixa qualidade são possivelmente atacados ou não objectivamente aceitáveis. Deste modo, a precisão é alta. Por outro lado, a abrangência representa a mais baixa abrangência dos mapeadores individuais. Além disso, um problema notável quando usando um valor de 1.0 refere-se a ausência de mapeamentos por um mapeador. Neste caso, se todos os outros mapeadores possuem mapeamentos verdadeiros positivos com confiança abaixo de 1.0, tais mapeamentos não atacados com sucesso pelos mapeamentos negativos.

Tabela 3: Diferentes valores de confiança para mapeamentos negativos – mapeadores OAEI.

	0.5	1.0
P^{macro}	0.70	**1.0**
P^{micro}	0.67	**1.0**
R^{macro}	**0.89**	0.13
R^{micro}	**0.93**	0.13
F^{macro}	**0.77**	0.40
F^{micro}	**0.78**	0.22

[3]in http://oaei.ontologymatching.org/2007/results/

Quando comparado com o *baseline* – Tabela 4 – os modelos de negociação e argumentação eliminam mapeamentos falsos positivos, resultando valores de precisão ligeiramente melhores que o baseline. Entretanto, significantes resultados são obtidos usando negociação. In termos de abrangência, não há significantes diferenças entre os resultados das três configurações.

Comparando negociação e argumentação, considerando que 0.5 é usado como confiança para mapeamentos negativos, precisão é baixa, enquanto abrangência é alta. Deste modo, negociação resulta melhores valores de precisão que argumentação, enquanto argumentação gera melhores valores de abrangência.

Tabela 4: Resultados de ***baseline***, negociação, e argumentação (confiança 0.5) – mapeadores OAEI.

	Baseline	Negociação	Argumentação
P^{macro}	0.61	**0.99**	0.70
P^{micro}	0.57	**0.99**	0.67
R^{macro}	**0.90**	0.79	**0.89**
R^{micro}	**0.94**	0.84	**0.93**
F^{macro}	0.71	**0.88**	0.77
F^{micro}	0.71	**0.91**	0.78

Quando analisando os resultados da negociação e argumentação com os resultados individuais dos mapeadores, o uso de negociação produz valores de precisão ligeiramente melhores que todos os mapeadores individuais. Com argumentação, um comportamento similar é encontrado para valores de abrangência. Em termos de F-measure, negociação produz resultados na faixa de valores dos melhores mapeadores individuais.

4 Caso Real: Biblioteca Nacional

Nesta configuração, dois grupos de mapeadores são usados. O primeiro grupo envolve três sistemas participantes da campanha de avaliação OAEI Library Track 2007: Falcon ([7], [11], [20]), DSSim ([18]), and Silas ([19]). O segundo grupo é formado por mapeadores alternativos, nomeadamente um sintáctico baseado em distância de edição, um lexical baseado em regras para mapeamento de termos em holandês, e um mapeador baseado na análise de instâncias das ontologias [8].

Para os experimentos usando argumentação, as ordens de preferências são as seguintes. Para o grupo dos mapeadores OAEI, por exemplo, a audiência do mapeador Falcon é a seguinte: Falcon > Silas > DSSim. Para os mapeadores alternativos, considerando o mapeador baseado em instâncias, a preferência é: SKOS > syntactic > co-occurrence. Para a combinação de todos os mapeadores, tem-se a preferência: Falcon > Silas > SKOS > syntactic > co-occurrence > DSSim. A confiança dos argumentos para mapeamentos negativos é de 1.0.

A Tabela 5 apresenta os resultados para o *baseline*, negociação, e argumentação para três combinações de mapeadores: grupo OAEI, grupo de mapeadores alternativos, e combinação de todos os mapeadores. Conforme esperado, negociação e argumentação eliminam mapeamentos

falsos positivos (e também alguns verdadeiros positivos), resultando em alta precisão, mas baixa abrangência, se comparados com os resultados do *baseline*.

Argumentação é mais selectiva. Quando um contra-argumento com confiança de 1.0 é gerado por um mapeador, ele ataca com sucesso os argumentos positivos dos mapeadores com menor preferência. Isto implica um incremento da precisão e um decremento da abrangência, quando comparada com o *baseline*. Para o grupo dos mapeadores OAEI (bem como a combinação que os envolve), a intersecção de mapeamentos é pequena (causada pelo baixo desempenho do sistema DSSim), o que implica na diminuição da abrangência. Considerando os mapeadores alternativos, a intersecção é grande, implicando em melhores valores de abrangência.

Usando negociação, é possível recuperar significante parte da intersecção dos conjuntos de mapeamentos, dada a selecção dos mesmos ser baseada em votos. Por exemplo, se ambos Falcon e Silas possuem um argumento a favor de um mapeamento positivo, independentemente da confiança de um possível argumento do sistema DSSim, o mapeamento positivo é seleccionado.

Tabela 5: Baseline, negociação, e argumentação.

	OAEI			Alternativos			Todos		
	P-a	R-a	J-a	P-a	R-a	J-a	P-a	R-a	J-a
Baseline	0.32	**0.46**	**0.26**	0.13	**0.79**	0.12	0.12	**0.80**	0.11
Negociação	**0.53**	0.32	**0.28**	0.45	0.44	**0.32**	**0.53**	0.38	**0.31**
Argumentação	0.52	0.07	0.07	**0.52**	0.37	**0.31**	**0.53**	0.07	0.06

Para o caso mais interessante, combinação de todos os mapeadores, precisão de ambos modelos de negociação e argumentação é similar. Entretanto, diferenças significantes são encontradas para a abrangência. De modo geral, negociação produz melhores resultados de J-a, os quais são similares ao melhor mapeador individual, melhorando significantemente os resultados dos piores mapeadores.

5 Conclusões e Trabalhos Futuros

Mapeamento de ontologias é visto como uma solução promissora para a heterogeneidade semântica, suportando interoperabilidade entre sistemas baseados em ontologias. Um aspecto importante nesta área refere-se a encontrar formas de escolha entre as diversas técnicas disponíveis e suas variações, e então combinar os seus resultados.

Nesta tese, o problema de combinar diferentes abordagens de mapeamento foi formalizado usando uma abordagem baseada em agentes cooperativos. Especificamente, dois modelos foram propostos: um modelo de negociação baseado em votos, e um modelo de argumentação baseado em confiança. em ambos os modelos, os mapeamentos são computador por agentes usando diferente abordagens de mapeamento. Usando o primeiro modelo, o consenso entre os agentes é representado pelo número de suportes para um mapeamento positivo ou negativo, onde o maior número é seleccionado como o consenso.

O modelo de argumentação foi baseado no *Value-based Argumentation Framework* (VAF). Tal modelo foi estendido de forma a representar argumentos com graus de confiança, de acordo

com o grau de similaridade entre as entidades sendo mapeadas. Uma nova noção de aceitabilidade foi definida, a qual combina valores (relacionados com a preferência dos agentes) e confiança dos argumentos. Baseado nas suas preferências e graus de confiança, os mapeadores computam seus conjuntos de mapeamentos preferíveis. Os argumentos em tais conjuntos são vistos como o conjunto de argumentos globalmente aceitáveis.

Usando argumentação é possível usar os valores para representar preferências entre os mapeadores. Cada abordagem representa uma audiência, com preferências entres os valores. Os valores são usados para determinar a preferência entre as diferentes abordagens. Além disso, cada agente gera argumentos com graus de confiança associados, de acordo com a medida de similaridade retornada pela técnica de mapeamento. Quando a performance dos mapeadores está disponível, uma ordem completa de preferência pode ser definida (i.e., A > B > C), enquanto quando esta informação não está disponível, a ordem parcial deve ser especificada (i.e., A > B; and A > C).

Diferentemente do modelo de argumentação, usando negociação um mapeador que não executa satisfatoriamente tem o mesmo peso no processo de voto que um mapeador que possui bons resultados. Uma relação de preferência simples é usada somente nos casos de empate nos votos, onde o mapeador com melhor desempenho decide o impasse (quando essa informação não está disponível, uma escolha arbitraria deve ser feita).

Uma vantagem potencial do modelo de argumentação é a possibilidade de ajustar preferências entre mapeadores. Por outro lado, o desempenho deste modelo está relacionado com os graus de confiança atribuídos aos mapeadores. É reconhecida a importância da associação de graus de confiança aos argumentos, reflectindo a confiança que o mapeador tem na similaridade entre entidades das ontologias. Tais graus de confiança são usualmente derivados a partir da avaliação de similaridade feita durante o processo de mapeamento. Entretanto, não existem teorias que suportem a definição de tais medidas. Usando estas medidas para comparar resultados de diferentes mapeadores é questionável. Por exemplo, um grau de confiança de 0.8 pode não corresponder ao mesmo nível de confiança para diferentes mapeadores. Além disso, o uso diferentes valores para representar a confiança de contra-argumentos implica em um custo-benefício entre precisão e abrangência. Tal evidência sugere a necessidade de um estudo mais detalhado de forma a especificar os valores a serem usados para balancear tais resultados.

Uma limitação do modelo de argumentação está relacionada ao facto de que um argumento contra um mapeamento pode atacar com sucesso todos os argumentos a favor deste mapeamento, independente do número de mapeamento a favor (especialmente quando um alto valor de confiança é usado para representar mapeamentos negativos). Por exemplo, três argumentos para um mapeamento verdadeiro positivo pode ser atacado com sucesso por um argumento representando um mapeamento falso negativo. Sob outra perspectiva, tal facto pode melhorar significantemente a precisão, enquanto reduzindo a abrangência.

Em média, negociação tem um desempenho melhor do que argumentação (especialmente para os valores de F–measure). Quando considerando a união de todos os mapeadores, ambos os modelos propostos apresentam resultados promissores.

Melhorar os resultados do mapeador com o maior desempenho é uma tarefa difícil, especialmente quando existem uma grande intersecção entre o conjunto de resultados. Neste caso, o desempenho dos modelos cooperativos é similar aos mapeadores individuais. Por outro lado,

quando os conjuntos são disjuntivos, os modelos cooperativos são promissores. Negociação tem provado ser útil dado o facto de que é baseada no número de vezes que um mapeamento é aceito, aumentando as chances deste ser válido. Quando usando argumentação com altos valores de confiança para contra-argumentos em mapeamentos negativos, significantes valores de precisão são obtidos. Deste modo, a combinação de ambas as abordagens poderia ser interessante, adaptando, por exemplo, o S-VAF para a introdução de votos na definição de ataques com sucesso; introduzindo graus de confiança ponderados, de acordo com o desempenho individual dos mapeadores; ou usando um meta-agente para combinar ambos os modelos.

Como trabalhos futuros, um modelo de argumentação baseado em votos poderia ser especificado, o qual considera o número de argumentos a favor ou contra um mapeamento. Como observado no processo de avaliação, quanto mais vezes um mapeamento é aceito, maior é a chance deste mapeamento ser válido. Segundo, um estudo quantitativo sobre o uso de diferentes valores de confiança para argumentos deve ser feito, de forma a avaliar as equivalências entre os graus de confiança retornadas pelos diferentes mapeadores. Terceiro, a meta-agente para combinar ambos os modelos de negociação e argumentação poderia ser especificado. Finalmente, outros conjuntos de dados deveriam ser usados para avaliar a qualidade das relações semânticas retornadas pelos mapeadores propostos.

Referências

[1] T. Bench-Capon. Persuasion in practical argument using value-based argumentation frameworks. *Journal of Logic and Computation*, 13(3):429–448, 2003.

[2] N. Choi, I.-Y. Song, and H. Han. A survey on ontology mapping. *ACM SIGMOD Record*, 35(3):34–41, 2006.

[3] C. Curino, G. Orsi, and L. Tanca. X-som: A flexible ontology mapper. In *Proceedings of the 18th International Conference on Database and Expert Systems Applications*, pages 424–428, Washington, DC, USA, 2007. IEEE Computer Society.

[4] C. Curino, G. Orsi, and L. Tanca. X-som results for oaei 2007. In *Proceedings of the 2nd Ontology Matching Workshop*, pages 276–285, Busan, Korea, 2007.

[5] J. Euzenat and P. Shvaiko. *Ontology Matching*. Springer-Verlag, Heidelberg (DE), 2007.

[6] J. Euzenat and P. Valtchev. Similarity-based ontology alignment in OWL-Lite. In *Proceedings of the European Conference on Artificial Intelligence*, volume 16, pages 333–337, 2004.

[7] W. Hu, N. Jian, Y. Qu, and Y. Wang. Gmo: A graph matching for ontologies. In B. Ashpole, M. Ehrig, J. Euzenat, and H. Stuckenschmidt, editors, *K-CAP Integrating Ontologies*, volume 156 of *CEUR Workshop Proceedings*, pages 43–50. CEUR-WS.org, 2005.

[8] A. Isaac, L. van der Meij, S. Schlobach, and S. Wang. An empirical study of instance-based ontology matching. In *The Semantic Web Conference 2007*, volume 4825/2008 of *Lecture Notes in Computer Science*, Busan, Korea, 2008. Springer Berlin / Heidelberg.

[9] Y. Jean-Mary and M. Kabuka. Asmov: Ontology alignment with semantic validation. In *Joint SWDB-ODBIS Workshop on Semantics, Ontologies, Databases*, pages 15–20, Vienna, Austria, 2007.

[10] Y. Jean-Mary and M. Kabuka. Asmov results for oaei 2007. In *Proceedings of the 2nd Ontology Matching Workshop*, pages 141–150, Busan, Korea, 2007.

[11] N. Jian, W. Hu, G. Cheng, and Y. Qu. Falconao: Aligning ontologies with falcon. In B. Ashpole, M. Ehrig, J. Euzenat, and H. Stuckenschmidt, editors, *K-CAP Integrating Ontologies*, volume 156 of *CEUR Workshop Proceedings*, pages 85–91. CEUR-WS.org, 2005.

[12] Y. Kalfoglou and M. Schorlemmer. Ontology mapping: The state of the art. *Knowledge Engineering Review*, 18(1):1–31, 2003.

[13] C.-C. Kiu and C.-S. Lee. Ontology mapping and merging through ontodna for learning object reusability. *Journal of Educational Technology and Society*, 9(3):27–42, 2006.

[14] C.-C. Kiu and C.-S. Lee. Ontodna: Ontology alignment results for oaei 2007. In *Proceedings of the 2nd Ontology Matching Workshop*, pages 196–204, Busan, Korea, 2007.

[15] P. Lambrix and H. Tan. Sambo: A system for aligning and merging biomedical ontologies. *Journal of Web Semantic*, 4(3):196–206, 2006.

[16] V. Levenshtein. Binary Codes Capable of Correcting Deletions and Insertions and Reversals. *Soviet Physics Doklady*, 10(8):707–710, 1966.

[17] M. Mao and Y. Peng. The prior+: Results for oaei campaign 2007. In *Proceedings of the 2nd Ontology Matching Workshop*, pages 219–226, Busan, Korea, 2007.

[18] M. Nagy, M. Vargas-Vera, and E. Motta. DSSim – managing uncertainty on the semantic web. In *2nd International Workshop on Ontology Matching*, Busan, Korea, 2007.

[19] R. Ossewaarde. Simple library thesaurus alignment with SILAS. In *2nd International Workshop on Ontology Matching*, Busan, Korea, 2007.

[20] Y. Qu, W. Hu, and G. Cheng. Constructing virtual documents for ontology matching. In *Proceedings of the 15th International Conference on World Wide Web*, pages 23–31, New York, NY, USA, 2006. ACM.

[21] E. Rahm and P. Bernstein. A survey of approaches to automatic schema matching. *International Journal on Very Large Data Bases*, 10(4):334–350, 2001.

[22] P. Shvaiko. A classification of schema-based matching approaches. Technical Report DIT-04-093, Informatica e Telecomunicazioni, University of Trento, 2004.

[23] P. Shvaiko and J. Euzenat. A survey of schema-based matching approaches. In *Journal on Data Semantics IV*, volume 3730/2005 of *Lecture Notes in Computer Science*, pages 146–171. Springer Berlin, 2005.

[24] V. Spiliopoulos, A. Valarakos, G. Vouros, and V. Karkaletsis. Sema: Results for the ontology alignment contest oaei 2007. In *Proceedings of the 2nd Ontology Matching Workshop*, pages 244–254, Busan, Korea, 2007.

[25] K. Sycara. Multiagent systems. *AI Magazine*, 10(2):79–93, 1998.

[26] H. Tan and P. Lambrix. Sambo results for the ontology alignment evaluation initiative 2007. In *Proceedings of the Second Ontology Matching Workshop*, pages 236–243, Busan, Korea, 2007.

[27] J. Tang, B. Liang, J.-Z. Li, and K. Wang. Risk minimization based ontology mapping. In *Advanced Workshop on Content Computing*, volume 3309 of *Lecture Notes in Computer Science*, pages 469–480. Springer Verlag, 2004.

[28] C. Trojahn, M. Moraes, P. Quaresma, and R. Vieira. Using cooperative agent negotiation for ontology mapping. In *Proceedings of the 4th European Workshop on Multi-Agent Systems (EUMAS)*, CEUR Workshop Proceedings. CEUR-WS.org.

[29] C. Trojahn, M. Moraes, P. Quaresma, and R. Vieira. A negotiation model for ontology mapping. In *Proceedings of the IEEE/WIC/ACM international conference on Intelligent Agent Technology*, pages 762–768, Washington, DC, USA, 2006. IEEE Computer Society.

[30] H. Wache, T. Vögele, U. Visser, H. Stuckenschmidt, G. Schuster, H. Neumann, and S. Hübner. Ontology-based integration of information: A survey of existing approaches. In A. Pérez, M. Gruninger, H. Stuckenschmidt, and M. Uschold, editors, *IJCAI Workshop on Ontologies and Information Sharing*, pages 108–117, Seattle, WA US, 2001.

[31] P. Wang and B. Xu. Lily: The results for the ontology alignment contest oaei 2007. In *Proceedings of the 2nd Ontology Matching Workshop*, pages 179–185, Busan, Korea, 2007.

[32] H. Zargayouna, B. Safar, and C. Reynaud. Taxomap in the oaei 2007 alignment contest. In *Proceedings of the 2nd Ontology Matching Workshop*, pages 268–275, Busan, Korea, 2007.

Abstract

Ontologies have proven to be an essential element in a range of applications in which knowledge plays a key role. Resolving the semantic heterogeneity problem is crucial to allow the interoperability between ontology-based systems. This makes automatic ontology matching, as an anticipated solution to semantic heterogeneity, an important research issue. Many different approaches to the matching problem have emerged from the literature. An important issue of ontology matching is to find effective ways of choosing among many techniques and their variations, and then combining their results. An innovative and promising option is to formalize the combination of matching techniques using agent-based approaches, such as cooperative negotiation and argumentation. In this thesis, the formalization of the ontology matching problem following an agent-based approach is proposed. Such proposal is evaluated using state-of-the-art data sets. The results show that the consensus obtained by negotiation and argumentation represent intermediary values which are closer to the best matcher. As the best matcher may vary depending on specific differences of multiple data sets, cooperative approaches are an advantage.

Resumo

Ontologias são elementos essenciais em sistemas baseados em conhecimento. Resolver o problema de heterogeneidade semântica é fundamental para permitir a interoperabilidade entre sistemas baseados em ontologias. Mapeamento automático de ontologias pode ser visto como uma solução para esse problema. Diferentes e complementares abordagens para o problema são propostas na literatura. Um aspecto importante em mapeamento consiste em selecionar o conjunto adequado de abordagens e suas variações, e então combinar seus resultados. Uma opção promissora envolve formalizar a combinação de técnicas de mapeamento usando abordagens baseadas em agentes cooperativos, tais como negociação e argumentação. Nesta tese, a formalização do problema de combinação de técnicas de mapeamento usando tais abordagens é proposta e avaliada. A avaliação, que envolve conjuntos de testes sugeridos pela comunidade científica, permite concluir que o consenso obtido pela negociação e pela argumentação não é exatamente a melhoria de todos os resultados individuais, mas representa os valores intermediários que são próximo da melhor técnica. Considerando que a melhor técnica pode variar dependendo de diferencas específicas de múltiplas bases de dados, abordagens cooperativas são uma vantagem.

To my family, specially my mother.

Acknowledges

Firstly, I thank my advisor Paulo Quaresma and my co-advisor, Renata Vieira, whose patience, motivation, and encouragement had allowed I could finish this thesis. They have helped me professional and personally. I can not forget to say very thanks to my master advisor Fernando Osorio.

I thank very specially my family for the unconditional support they provided me through my entire life and in particular in these three last years. I would like to thank also the special support of my friends, specially Cris, Marcirio, Paula, Alexandre, Euler, Tiago, Lu, and Ana Luisa.

I appreciate the attention of the members of my committee for taking time out from their busy schedules to contribute to my thesis.

A special thanks goes out to Arno Lodder and Antoine Isaac for receiving me at Department of Law and Intelligence Artificial at Vrije Universiteit, Holland, during my internship.

I thank also CENTRIA (Artificial Intelligence Center of the New University of Lisbon) for supporting my registration in some international conferences.

Finally, I recognize that this research would not have been possible without the financial support of Programme AlBan, the European Union Programme of High Level Scholarships for Latin America, scholarship no E05D059374BR. I express my gratitude to this agency.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The term *ontology* has its origin from Philosophy, where it means the study of the organization of the reality. In Computer Science, Ontology is a name for an explicit representation of the knowledge about a domain. Researches on ontologies have received attention in Artificial Intelligence (AI) community, specially in the areas of Knowledge Engineering, Natural-Language Processing, Information Retrieval and more recently the Semantic Web. The reason ontologies are becoming popular is due to what they promise: a shared and common understanding of some domain that can be communicated between people and application systems (Fensel [2003]). They have proven to be an essential element in a range of applications in which knowledge plays a key role.

The ontology engineering process involves different designers with different views of the reality. The resulting ontologies involve different conventions, granularity, and coverage. Heterogeneity in the applications using these ontologies arise in different levels, such as (Euzenat and Shvaiko [2007]): (i) *syntactic heterogeneity*, occurring when two ontologies are not expressed in the same ontology specification language; (ii) *terminological heterogeneity*, which occurs due to variations in names when referring to the same entities in different ontologies; (iii) *conceptual heterogeneity*, which stands for the differences in modeling the same domain of interest; and (iv) *semiotic heterogeneity*, which is concerned with how entities are interpreted by people. Semantic heterogeneity is assumed to comprises both the *terminological* and *conceptual* heterogeneities.

Resolving the semantic heterogeneity problem is crucial to allow the interoperability between ontology-based systems. This makes automatic ontology matching, as an anticipated solution to semantic heterogeneity, an important research issue.

Ontology *matching* is the process of finding relationships or correspondences between entities of different ontologies. The output of this process is an *alignment*: a set of correspondences between two or more ontologies. A *correspondence* (or a *mapping*) is the relation holding, according to a particular matching algorithm, between entities of different ontolo-

gies. The matching process is often refereed as ontology mapping or ontology alignment (e.g., as in Choi et al. [2006]).

Ontology matching applications range from traditional applications such as ontology engineering and information integration to more recently agent communication, query answering, and navigation on the Semantic Web. In these applications, imposing a central, common ontology is not realistic. So, matching of heterogeneous ontologies is an intrinsic problem. For instance, ontology engineering involves to deal with multiple and distributed ontologies, which often need to be put together. In information integration, the basic idea is to provide users with a unified view of different and local information sources, usually on the basis of a global information view over which queries can be expressed. This involves to identify correspondences between semantically related entities in the local sources.

In multi-agent communication, ontologies play a fundamental role, formalizing the vocabulary from the agent's perception of the world and agents using different ontologies need to agree on the vocabulary they use, in order to communicate and then resolve their tasks. Ontology matching is a primary problem that has to be solved in order to allow agents with different backgrounds to adjust themselves before starting any form of cooperation or communication. The navigation and query answering on the web are other scenarios where matching is required, specially on the semantic web, where ontologies are proposed to be used to describe the content of the available resources. In the navigation scenario, the content of web pages can be annotated with term of ontologies and searching on the web involves matching between the ontologies describing these content (Sabou et al. [2006]).

1.1 Motivation

Many different approaches to the matching problem have emerged from the literature. The distinction between them is accentuated by the manner in which they exploit the features within an ontology. Whereas syntactic approaches consider measures of string similarity; semantic ones consider semantic relations usually on the basis of semantic oriented linguistic resources. Other approaches consider term positions in the ontology hierarchy or instances of the ontologies (Isaac et al. [2008b]). However, each category offers a wide diversity of options. Such approaches have been surveyed from different perspectives in (Rahm and Bernstein [2001]), (Wache et al. [2001]), (Kalfoglou and Schorlemmer [2003b]), (Shvaiko [2004]), (Shvaiko and Euzenat [2005]), (Choi et al. [2006]), and (Euzenat and Shvaiko [2007]).

The matching systems are based on the assumption that using a single technique is not sufficient to the problem and different matching techniques are aggregated in an unified process. Such aggregation involves parallel or sequential execution of the matchers and can vary from single weighted sum of the individual matcher results to learning the best matcher

or the combination of them.

Moreover, some techniques will perform better than others for specific cases, depending on how well the technique fits the material available as well as approaches that perform well for a specific case can not be successful in other ones. An important issue of ontology matching is to find effective ways of choosing among many techniques and their variations, and then combining their results.

An innovative and promising option is to formalize the combination of matching techniques using agent-based approaches, such as cooperative negotiation and argumentation. Following such perspective, different agents work on the basis of particular approaches arriving to distinct matching results that must be shared, compared, chosen and agreed. Such approaches allow to combine matching approaches in a more effective way. First, the final result of negotiation and argumentation represent a *consensus* between different views. Second, solving conflicts, the individual results can be improved. Third, agents can be aggregated and replaced in order to represent more interesting approaches according to the specific case, allowing to adapt the configuration of the matching solution.

1.2 Objectives

This thesis aims to formalize the problem of combination of ontology matching approaches using a cooperative agent-based approach. The objective is to use approaches from Multi-agent Systems (MAS) research to matching problem, and to investigate which approach is more suitable to the problem. The following specific objectives are defined:

1. Review the state-of-the-art on ontology matching;
2. Specify individual matcher agents;
3. Review the state-of-the-art on cooperative negotiation;
4. Review the state-of-the-art on argumentation;
5. Specify a cooperative negotiation model to combine ontology matching approaches;
6. Specify an argumentation model to combine matching approaches;
7. Evaluate the individual matchers;
8. Evaluate the negotiation and argumentation models;
9. Compare the proposed models against state-of-the-art matching systems.

1.3 Proposed Approach

Researches in Multi-agent Systems (MAS) are motivated by the conceptualization, design, and implementation of entities (agents) that operate in a distributed and open environment. The key characteristics of MAS are (Sycara [1998]) that (i) each agent has incomplete information or capabilities for solving the problem and, thus, has a limited viewpoint; (ii) there is no system global control; (iii) data is decentralized; and (iv) computation is asynchronous.

In order to solve conflicts that arise when several agents need to cooperate and coordinate their activities, mechanisms of negotiation and argumentation are used. Both approaches refer to some kind of communication between agents. However, systems involving negotiation and argumentation are very different in their nature ranging from communication in form of auctions to argumentation in a more philosophical sense.

Negotiation involves the communication between agents that iteratively exchange proposals and counterproposals. Argumentation can be seen as a more sophisticated exchange of deals in a negotiation protocol or a model for reasoning based on the construction and comparison of arguments. In the first sense, arguments are seen as meta-information used to justify the negotiation proposals or to persuade agents to change their negotiation stance. In the second sense, argumentation can be abstractly defined as the interaction of different arguments for and against some conclusion.

Both negotiation and argumentation are applied to the matching problem. In the negotiation model the agents interact following a voting-based protocol where each proposal represents a vote in favor or against a mapping between terms of the ontologies. Thus, the consensus is voting-based. Basically, the negotiation process involves two phases. First, the agents work in an independent manner, applying a specific mapping approach and generating a set of negotiation objects. Second, the set of negotiation objects is negotiated among the agents. The negotiation process involves one mediator and several matcher agents.

Using argumentation, mappings are represented as arguments. According to the definition of attacking relations, an argument for a mapping generated by one matcher can be supported or attacked by other arguments from other matchers. Based on the argumentation framework instantiation (using specific attacking relation and preference order between the arguments), the matchers compute their preferred set of arguments. The arguments in such preferred sets are viewed as the set of globally acceptable arguments (mappings). The framework which is used to deal with consensus, is Strength Value-based Argumentation Framework (S-VAF), a proposed extension to Value-based Argumentation Framework (VAF) by (Bench-Capon [2003]). The objective of the S-VAF is to consider the strength of an argument when defining the relation of acceptability.

Value-based Argumentation acknowledges the importance of preferences when considering arguments. However, in the specific context of ontology matching, an objection can still be

raised about the lack of complete mechanisms for handling persuasiveness. Indeed, many matching tools actually output mappings with a strength that reflects the confidence they have in the similarity between the two entities. These confidence levels are usually derived from similarity assessments made during the alignment process.

A detailed evaluation of both negotiation and argumentation models is done using state-of-the-art case studies, which are refereed by the ontology matching community. A comparison between both models is done, as well as the their results are evaluated against state-of-the-art ontology matching systems.

1.4 Main Contributions

The main contributions of this thesis are:

- Formalizing the problem of combining ontology matching techniques using agent-based approaches;
- Specifying a cooperative negotiation model based on voting to combine ontology matching approaches;
- Specifying a Strength Value-based Argumentation Framework (S-VAF) in order to deal with an important issue in ontology matching, namely the confidence of the mappings;
- Specifying matchers able to output different semantic relations than the traditional *equivalence*, namely narrower and broader relations;
- Combining and evaluating matchers based on different matching strategies (namely syntactic, semantic, and structural);
- Evaluating the weakness and strongness of using agent-based approaches to combine matching approaches.

1.5 Organization of the Thesis

The thesis is organized as follows. Chapter 2 presents a review on ontologies and ontology matching. The ontology matching techniques and strategies to combine them are commented. In Chapter 3, negotiation and argumentation approaches are presented, where argumentation is both considered as a sophisticated sub-class of the negotiation approaches and a reasoning model to deal with different levels of the comparison of arguments. Chapter

4 presents a comparative study of the main ontology matching systems proposed in the literature is are presented. The systems using negotiation and argumentation are detailed. Chapter 5 presents the proposed *cooperative negotiation* and *argumentation* models. Three proposed matcher agents representing individual matching techniques are described. In Chapter 6, the evaluation of the proposed models are presented. Two data sets are used: (i) a benchmark of ontologies in the bibliographic domain provided by the Ontology Alignment Evaluation Initiative (OAEI); and (ii) a real-world library case, from National Library of the Netherlands. Finally, Chapter 7 presents the concluding remarks and future work.

Chapter 2

Ontology Matching

Ontologies provide a formal description of the objects and their relations in a domain. They have a key importance for applications such as information retrieval, database integration, peer-to-peer systems, e-commerce, or semantic web services. In information retrieval systems, the terms defined in an ontology are used as metadata to markup and these semantic markups are semantic index terms for information retrieval, in order to improve the information retrieval (Jun-feng et al. [2005]). Other typical application is question answering, where the data sources used to construct the answer are described using populated ontologies instead of databases. Tasks in both these systems are likely to require support from more than one ontology for obvious reasons, and ontology matching is required for that. In a question answering system, such matching process can be used to define mappings between the ontologies on the fly to describe data sources available to construct an answer.

Ontology matching process takes two ontologies as input and determines as output correspondences between the semantically related entities of those ontologies. There are several matching approaches related to different aspects of concepts similarity. Different ontology mapping approaches are required, as terms may be mapped by a measure of lexical similarity (Stoilos et al. [2005]), (Maedche and Staab [2002]), or they can be evaluated semantically, usually on the basis of semantic oriented linguistic resources, or considering the term positions in the ontology hierarchy (Hakimpour and Geppert [2001]). It is assumed that the approaches are complementary to each other and combining different ones reflect better solutions when compared to the solutions of the individual approaches.

In this chapter, a review on ontologies and ontology matching is presented. The chapter is organized as follows. Section 2.1 comments on ontologies and the ontologies specification languages. Section 2.2 presents an overview of ontology matching applications, techniques and strategies. Finally, Section 2.3 presents the summary of the chapter in the context of the thesis.

2.1 Thesauri, Schemas and Ontologies

In Computer Science, Ontology is a name for an explicit representation of the knowledge about a domain. Researches on ontologies have received attention in Artificial Intelligence (AI) community, specially in the areas of Knowledge Engineering, Natural-Language Processing, Information Retrieval and more recently the Semantic Web. The reason ontologies are becoming popular is due to what they promise: a shared and common understanding of some domain that can be communicated between people and application systems (Fensel [2003]). They have proven to be an essential element in a range of applications in which knowledge plays a key role.

There are many vocabulary data structures and conceptual models that share similarities with ontologies. These are, for instance, folksonomies (simple tags used to content annotation), database schemas (typically organized into tables), directories (hierarchy of folders identified by labels and containing items, such as Google[1] and Yahoo[2] directories), thesauri (a hierarchical model containing other relations such as synonymous and antonymous), and ontologies (which are supposed to have an explicit well defined semantic).

Although ontologies have a similar function as a database schema, (Fensel [2003]) point out the main differences: a language for defining ontologies is syntactically and semantically richer than common approaches for databases; an ontology must be a shared and consensual terminology because it is used for information sharing and exchange; an ontology provides a domain theory and not the structure of a data container.

The distinctive feature of ontologies is the existence of a model theoretic semantic: ontologies are logic theories (Euzenat and Shvaiko [2007]). The semantic provides the rules for interpreting the syntax, which do not provide the meaning directly but constrains the possible interpretations of what is declared.

In this section, ontologies are discussed in more detail.

2.1.1 Ontologies

The term *ontology* has its origin from Philosophy, where it means the study of the organization of the reality. In AI, the term is used to refer to a body of knowledge describing some domain, using a representation vocabulary (Chandrasekaran et al. [1999]). Such vocabulary provides a set of term which is used to describe the facts in some domain, while the body of knowledge using that vocabulary is a collection of facts about the domain. From this broad meaning, different definitions of what is an ontology are proposed in the literature. (Guarino

[1] http://www.google.com/dirhp
[2] www.yahoo.com

and Giaretta [1995]) presents a range of interpretations to the term:

1. ontology as a philosophical discipline.
2. ontology as an informal conceptual system.
3. ontology as a formal semantic account.
4. ontology as a specification of a conceptualization.
5. ontology as a representation of a conceptual system via a logical theory (characterized by specific formal properties).
6. ontology as the vocabulary used by a logical theory.
7. ontology as a (meta-level) specification of a logical theory.

The interpretation 4, proposed by (Gruber [1993]), has been considered the most intuitive one to define an ontology: "a formal, explicit specification of a shared conceptualization". From this definition, (Fensel [2003]) point out that: *conceptualization* refers to an abstract model of some phenomenon in the world which identifies the relevant concepts of that phenomenon – from (Gruber [1995]), a conceptualization is an abstract, simplified view of the world that it is wished to represent for some purpose); *explicit* means that the type of concepts used and the constraints on their use are explicitly defined; *formal* refers to the fact that the ontology should be machine readable – the ontology is supposed to be formal: the notions it captures are thus precise and unambiguous (Bench-Capon [2005]); and *shared* reflects the notion that an ontology captures consensual knowledge, that is, it is not restricted to some individual, but accepted by a group.

An ontology may take a variety of forms, but necessarily it will include a vocabulary of terms, and some specification of their meaning. This includes definitions and an indication of how concepts are inter-related which collectively impose a structure on the domain and constrain the possible interpretations of terms (Uschold [1998]). Formally, an ontology is the statement of a logical theory (Gruber [1993]). Ontologies are content theories about the sorts of objects, proprieties of objects, and relations between objects that are possible in a specified domain of knowledge (Chandrasekaran et al. [1999]). In some cases, ontologies denote the result of activities like conceptual analysis and domain modeling (Guarino [1998]).

Although differences exist within ontologies, general agreement exists between ontologies on many issues (Chandrasekaran et al. [1999]):

- there are *objects* in the world.
- objects have *properties* or *attributes* that can take *values*.

- objects can exist in various *relations* with each other.
- properties and relations can change over *time*.
- there are *events* that occur at different *time* instants.
- there are *processes* in which objects participate and that occur over time.
- the world and its objects can be in different *states*.
- events can *cause* other events or states as *effects*.
- objects can be have *parts*.

Basically, ontologies express the types of *objects* in the domain; the *attributes* which these objects may have; the *relationships* which these objects may enter into; and the *values* that the attributes may have for particular types. An ontology together with a set of individual instances of classes constitutes a knowledge base (Noy and McGuiness [2001]).

Ontologies range in abstraction, from very general terms that form the foundation for knowledge representation in all domains, to terms that are restricted to specific knowledge domains (Chandrasekaran et al. [1999]). Depending on their generality level, different types of ontologies may be identified (Bench-Capon [2005]):

- *lightweight ontologies*: consists of a set of hierarchically organized terms. Typically, the purpose of such ontologies is to assist in information retrieval.
- *upper or top ontologies* (generic or common sense ontologies): attempts to describe fundamental categories applicable to all domains. Events, individuals, relations and concepts relating to time and action are found in such an ontology. Well-known upper ontologies include Dublin Core, Cyc[3], SUMO (Suggest Upper Merged Ontology) (Niles and Pease [2001]), and DOLCE (Descriptive ontology for Linguistic and Cognitive Engineering) (Masolo et al. [2003]).
- *core or domain ontologies*: attempts to articulate the concepts fundamental to some particular domain. Biology is a typical domain for ontologies, many have been proposed on different aspects of the field.
- *application ontologies* (method or task ontologies): contains the very detailed and specific concepts required to perform a particular task on a particular piece of an application.

[3]http://www.cyc.org

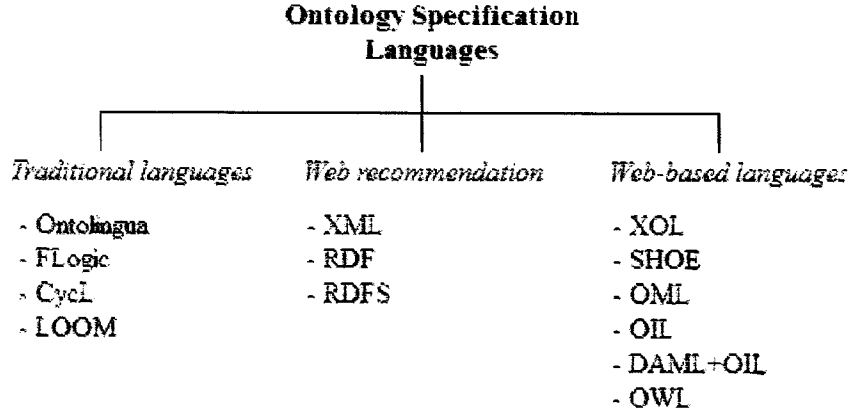

Figure 2.1: Ontology related languages.

2.1.1.1 Ontology Specification Languages

Ontologies may be expressed in different languages. These languages are characterized by different expressiveness, facilities, and syntaxes.

Following the classification proposed by (Óscar Corcho and Gómez-Pérez [2000], Gomez-Perez and Corcho [2002]), the ontology languages can be grouped into three categories: *traditional ontology languages*, *web standard and recommendations*, and *web-based ontology languages*. Figure 2.1 shows a schematic classification.

The *traditional ontology languages* comprise languages such as the well-known Ontolingua (Gruber [1993]), a language based on an extended version of first-order predicate language KIF (Knowledge Interchange Format), and it is used as basis to the translation to multiple other representation languages (e.g., Loom and CycL); FLogic (Frame Logic) (Kifer et al. [1995]), a declarative representation language that integrates frame-based languages and first-order predicate calculus; CycL, a language based on first-order predicate calculus with some higher-order extensions, created in the context of the Cyc Project; and LOOM (MacGregor [1991]), a language based on Description Logics (DL) (Baader et al. [2003]).

The second group comprises web standards, created in the context of the Semantic Web. XML (eXtended Markup Language)[4] is a development of the World Wide Web Consortium (W3C), which provides a surface syntax for structured documents, but imposes no semantic constraints on the meaning of these documents. RDF (Resource Description Framework)[5] was created to specify the semantics for data based on XML, in a standardized manner. The goal of RDF is to define a mechanism for describing resources that makes no assumptions about a particular application domain nor the structure of a document containing information.

[4]http://www.w3.org/TR/REC-xml/
[5]http://www.w3.org/RDF

RDF is a datamodel for objects ("resources") and relations between them and provides a simple semantics for this datamodel. The RDF data model does not provide mechanisms for describing the relationships between properties and resources, which is provided by the RDF Schema (RDFS)[6]. RDFS is a declarative language based on frames, used for the definition of RDF schemas. It is a vocabulary for describing properties and classes of RDF resources, with a semantics for hierarchies of proprieties and classes.

The third group, *web-based ontology specification languages*, comprises languages such as XOL (XML-Based Ontology Exchange Language)[7], SHOE (Simple HTML Ontology Extension)[8], OML (Ontology Markup Language)[9], OIL (Ontology Interchange Language)[10] (Fensel et al. [2000]), DAM-OIL (DARPA Agent Markup Language – Ontology Interchange)[11], and OWL (Ontology Web Language)[12]. XOL was designed to provide a format for exchanging ontology definitions among heterogeneous systems in the Bioinformatics community. It is not intended to be used for the development of ontologies, but as an intermediate language based on XML for transferring ontologies among different database systems. SHOE has been developed to incorporate semantic knowledge in HTML or other WWW documents. OML is partially based on SHOE, with conceptual graphs features to represent concepts and their relations. OIL is a proposal for a joint standard for describing and exchanging ontologies. It has a syntax and a semantic based on XOL and RDFS, providing modeling primitives used in frame-based approaches and formal semantic and reasoning support from Description Logics. DAML+OIL is a language for describing Web resources, based on RDF and RDF Schema. It extends these languages with modelling primitives found in frame-based languages (like OIL) DAML+OIL was built from the original DAML ontology language.

More recently, OWL, which was developed by the W3C, has been considered the state-of-the-art ontology language. OWL is a language for making ontological statements, developed based on RDF and RDFS, as well as earlier ontology languages including OIL and DAML+OIL. OWL adds more vocabulary for describing proprieties and classes.

An abstract syntax for OWL can be described using Description Logics. In the context of this thesis, the ontologies are assumed to be represented in OWL. In the rest of this section, Description Logics (DL) are described as a modeling language for OWL ontologies. Next, the OWL language is detailed.

Description Logics

[6]http://www.w3.org/TR/2000/CR-rdf-schema-20000327/
[7]http://www.ai.sri.com/pkarp/xol/xol.html
[8]http://www.cs.umd.edu/projects/plus/SHOE/
[9]http://www.ontologos.org/OML/OML200.3.htm
[10]http://www.ontoknowledge.org/oil
[11]http://www.w3.org/TR/daml+oil-reference
[12]http://www.w3.org/TR/owl-ref

Description Logics (DLs) are a family of knowledge representation (KR) formalisms that represent the knowledge of an application domain ("the world") by first defining the relevant concepts of the domain (its terminology), and then using these concepts to specify properties of objects and individuals occurring in the domain ("the world description") (Baader et al. [2003]).

A knowledge base (KB) based on DL represents *intensional* knowledge, which specifies the vocabulary of an application domain in the form of a terminology (TBox), and *extensional* knowledge, that is specific to the individuals of the domain discourse (ABox). The vocabulary consists of *concepts* and *roles*. *Concepts* denote set of individuals while *roles* denote binary relationship between individuals. Atomic (elementary) concepts and roles can be used to build complex description, using *constructors*. The language for building descriptions is a feature of different DLs, and different systems are distinguished by their description languages, i.e., the expressiveness of the language according with the *constructors* that they support.

The DLs are equipped with a formal, logic-based semantic, and reasoning mechanisms, that allow to infer implicity knowledge from the knowledge explicitly represented in the knowledge database. The basic reasoning tasks for DL systems are to determine whether a description is more general than another one, that is, whether the first subsumes the second (*subsumption*); whether a description is *satisfiable* (i.e., non-contradictory); or whether its set of assertions is *consistent*, that is, whether it has a model, and whether the assertions in the terminology entail that a particular individual is an *instance* of a given concept description.

SHOIN(D) DL (Horrocks and Patel-Schneider [2003]) can be seen as a standard DL language. It provides constructors for full negation, disjunction, a restricted form of existential quantification, and reasoning with concrete datatypes. The set of *SHOIN(D)* concepts is defined by the following syntactic rules, where A is an atomic concept, R is a role name, d is a concrete domain, c_i are individuals, and n is a non-negative integer:

$C \rightarrow$	A	|	(atomic concept)
	$\top$	|	(universal concept)
	$\bot$	|	(bottom concept)
	$\neg C$	|	(full negation)
	$C_1 \sqcap C_2$	|	(intersection)
	$C_1 \sqcup C_2$	|	(union)
	$\exists R.C$	|	(existential quantification)
	$\forall R.C$	|	(value restriction)
	$\geq n\, R$	|	(cardinality restriction)
	$\leq n\, R$	|	(cardinality restriction)
	$\{c_1,...,c_n\}$	|	(one of)
$D \rightarrow$	d	|	(concrete domain)

For instance, supposing that Person and Male are atomic concepts, Person $\sqcap$ Male and Person $\sqcap \neg$ Male are *SHOIN(D)* concepts, describing those persons that are male, and those that are not male, respectively.

The existential restrictions specifies the existence of a (i.e. at least one) relationship along a given property to an individual that is member of a specific class, while the universal restrictions constrain the relationships along a given property to individuals that are members of a specific class. Considering hasChild as an atomic role, Person $\sqcap \exists$hasChild.$\top$ and Person $\sqcap \forall$hasChild.Male are concepts denoting those persons that have a child and those persons whose children are all male. The bottom concept is used to describe, for example, those persons without child (Person $\sqcap \exists$hasChild.$\bot$).

The cardinality restrictions describe classes of individuals that have at least, at most or exactly a number of relationships with other individuals or datatypes. For a given property P, a minimum cardinality restriction specifies the minimum number of P relationships that an individual must participate in. A maximum cardinality restriction specifies the maximum number of P relationships that an individual can participate in. A cardinality restriction specifies the exact number of P relationships that an individual must participate in. For instance, the concept ($\geq$ 3 hasChild) $\sqcap$ ($\leq$ 2 hasFemaleRelative) represents the concept of individuals having at least three children an at most two female relatives.

The concrete domains includes, basically, data types, such as numbers and strings. It allows to integrate numerical and other domains in a schematic way into DLs (Baader et al. [2003]). An example of such a concrete domain is the set of nonnegative integers, with predicates such as $\geq$ and $\leq$. For example, an adequate definition of the concept Woman could consider a female that is old enough: Woman = Human $\sqcap$ Female $\sqcap \exists$has-age$\geq$.18, where $\geq$.18 stands for the unary predicate n | n $\geq$ 18 of all nonnegative integers greater than or equal to 18.

The semantics of the *SHOIN(D)* concepts is given by the *interpretation I* that consist of a non-empty set Δ^I, the domain of interpretation, and an interpretation function, which assigns to every atomic concept A a set $A^I \subseteq \Delta^I$, and to every atomic role $R^I \subseteq \Delta^I$ x Δ^I. The interpretation function is extended to concept description by the following inductive definitions:

```
<owl:Class rdf:about="#Woman">
  <owl:equivalentClass>
    <owl:Class>
      <owl:intersectionOf rdf:parseType="Collection">
        <owl:Restriction>
          <owl:onProperty>
            <owl:FunctionalProperty rdf:about="#hasSex"/>
          </owl:onProperty>
          <owl:hasValue rdf:resource="#Female"/>
        </owl:Restriction>
        <owl:Class rdf:about="#Person"/>
      </owl:intersectionOf>
    </owl:Class>
  </owl:equivalentClass>
</owl:Class>
```

Figure 2.2: Fragment of OWL code.

$$
\begin{array}{lcl}
A & = & A^I \subseteq \Delta^I \\
\top^I & = & \Delta^I \\
\bot^I & = & \emptyset \\
(\neg C)^I & = & \Delta^I \setminus C^I \\
(C_1 \sqcap C_1)^I & = & C_1{}^I \cap C_2{}^I \\
(C_1 \sqcup C_2)^I & = & (C_1 \cup C_2)^I \\
(\exists R.C)^I & = & \{a \in \Delta^I \mid \exists b.(a,b) \in R^I \wedge b \in C^I\} \\
(\forall R.C)^I & = & \{a \in \Delta^I \mid \forall b.(a,b) \in R^I \rightarrow b \in C^I\} \\
(\geq n R^I & = & \{a \in \Delta^I \| \mid \{b \mid \in R^I\} \mid \geq n\} \\
(\leq n R^I & = & \{a \in \Delta^I \| \mid \{b \mid \in R^I\} \mid \leq n\} \\
D^D & = & D^D \subseteq \Delta^I
\end{array}
$$

Concepts and roles are related by *terminological axioms*. Basically, there are two types of terminological axioms: *inclusion* and *equality*. The inclusion axioms have a form $C \sqsubseteq D$ or $R \sqsubseteq S$, where C and D are concepts, and R and S are roles. The second kind of axioms have a form $C \equiv D$ or $R \equiv S$. An equality whose left-hand side is an atomic concept is a *definition*. Definitions are used to introduce *symbolic names* for complex descriptions. For instance, the axiom Father $\equiv$ Man $\sqcap$ $\exists$hasChild.Person describes the concept Father. Symbolic names may be used as abbreviations in other description, e.g., Parent $\equiv$ Mother $\sqcup$ Father.

Ontology Web Language (OWL)

OWL is a XML syntax for DL. Basically, an OWL ontology consists of classes, properties, and individuals (instances of classes). Figure 2.2 shows an example of OWL code, which describes the concept *Woman* as a person with the property *hasSex* with type *Female* (Woman = Person $\sqcup$ $\forall$hasSex.Female).

OWL classes are a concrete representation of concepts. They are described using formal

descriptions that state precisely the requirements for membership of the class (Horridge et al. [2004]). Classes may be organized into super-class and sub-class hierarchy (subsumption relation).

OWL properties are used to describe a class. Properties have a domain and a range specified (they link individuals from the domain to individuals from the range) and can be linked to concrete datatypes or objects. Moreover, they may have sub properties, in order to form hierarchies of properties (for example, the property hasMother might specialize the more general property hasParent). Other relations between properties are:

- inverse properties: object properties can have a corresponding inverse property. If some property links individual a to individual b then its inverse property will link the individual b to the individual a.
- functional properties: if a property is functional, for a given individual, there can be at most one individual that is related to the individual via the property.
- inverse functional properties: If a property is inverse functional then it means that the inverse property is functional.
- transitive properties: If a property is transitive, and the property relates the individual a to the individual b, and also individual b to the individual c, then it can be inferred that the individual a is related to the individual c via the property P.
- symmetric properties: If a property P is symmetric, and the property relates the individual a to the individual b then the individual b is also related to the individual a via the property P.

OWL ontologies may be categorized into three sub-languages (Horridge et al. [2004]): OWL-Lite, OWL-DL and OWL-Full. A defining feature of each sub-language is its expressiveness. OWL-Lite is the least expressive sub-language. OWL-Full is the most expressive sub-language. The expressiveness of OWL-DL falls between that of OWL-Lite and OWL-Full. OWL-DL may be considered as an extension of OWL-Lite and OWL-Full an extension of OWL-DL.

OWL-Lite is the syntactically simplest sub-language. It is intended to be used in situations where only a simple class hierarchy and simple constraints are needed. For example, while OWL Lite supports cardinality constraints, it only permits cardinality values of 0 or 1. It should be simpler to provide tool support for OWL Lite than its more expressive relatives, and provide a quick migration path for thesauri and other taxonomies. The OWL Lite supports the following elements: schema features (class, property, subClassOf, subPropertyOf, domain, range, individual); equality and inequality (sameClassAs, samePropertyAs,

sameIndividualAs, differentIndividualFrom); property characteristics (inverseOf, transitiveProperty, symmetricProperty, unique functionalProperty, inverseFunctionalProperty, allValuesFrom – universal local range restrictions, someValuesFrom – existential local range restrictions); restricted cardinality (minCardinality – restricted to 0 or 1, maxCardinality – restricted to 0 or 1).

OWL-DL is more expressive than OWL-Lite and is so named due to its correspondence with Description Logics. It is therefore possible to automatically compute the classification hierarchy and check for inconsistencies in an ontology that conforms to OWL-DL. One of the key features of OWL-DL is that these relationships can be computed automatically by a reasoner. OWL DL supports those users who want the maximum expressiveness without losing computational completeness (all entailments are guaranteed to be computed) and decidability (all computations will finish in finite time) of reasoning systems. OWL DL includes all OWL language constructs with restrictions such as type separation (a class can not also be an individual or property, a property can not also be an individual or class). OWL DL was designed to support the existing Description Logic business segment and has desirable computational properties for reasoning systems. It supports the following elements, in addition the elements supported by the OWL-Lite: class axioms (oneOf – enumerated classes, disjointWith, sameClassAs applied to subClassOf), boolean combinations of class expressions (unionOf, intersectionOf, complementOf), arbitrary cardinality, and filler information (hasValue that can include specific value information).

OWL-Full is the most expressive OWL sub-language. It is intended to be used in situations where very high expressiveness is more important than being able to guarantee the decidability or computational completeness of the language. OWL Full is meant for users who want maximum expressiveness and the syntactic freedom of RDF with no computational guarantees. For example, in OWL Full a class can be treated simultaneously as a collection of individuals and as an individual in its own right. Another significant difference from OWL DL is that a owl:DatatypeProperty can be marked as an owl:InverseFunctionalProperty. OWL Full allows an ontology to augment the meaning of the pre-defined (RDF or OWL) vocabulary. It is unlikely that any reasoning software will be able to support every feature of OWL Full.

2.2 Ontology Matching

2.2.1 Ontology Matching, Translation, and Merging

In Ontology Matching community different terminologies are used to refer *matching*, *mapping*, *alignment*, *integration*, *merging*, and *translation*. In this section, the interpretations of these terms, which are adopted in the context of this thesis, are presented.

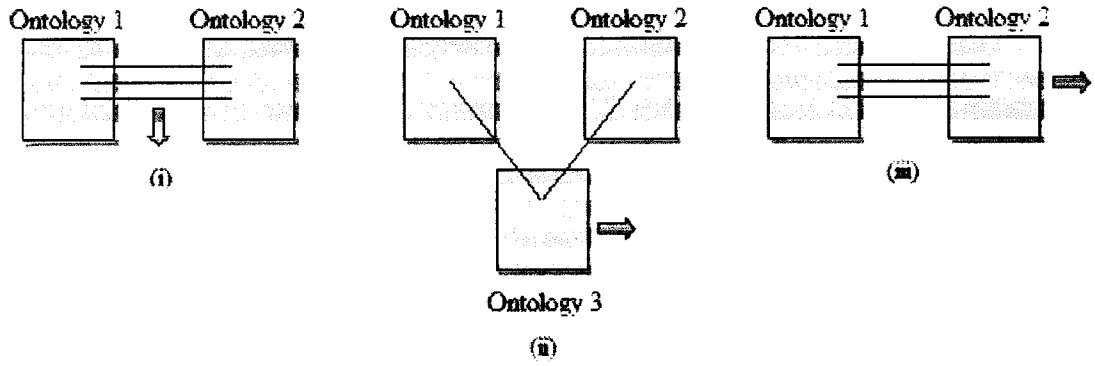

Figure 2.3: (i) ontology matching, (ii) merging, and (iii) integration.

Ontology *matching* (Figure 2.3) is the process of finding relationships or correspondences between entities of different ontologies. The output of this process is an *alignment*: a set of correspondences between two or more ontologies. A *correspondence* (or a *mapping*) is the relation holding, according to a particular matching algorithm, between entities of different ontologies. The matching process is often refereed as ontology mapping or ontology alignment (e.g., as in Choi et al. [2006]).

There are different definitions of mapping. According to (Euzenat and Shvaiko [2007]), a *mapping* is the oriented, or directed, version of an alignment: it maps the entities of one ontology at most one entity of another ontology. This complies with the mathematical definition of a mapping instead of that of a general relation. The mathematical definition would in principle require that the mapped object is equals to its image, i.e., that the relation is an equivalence relation. A mapping can be seen as a collection of mapping rules all oriented in the same direction, i.e., from one ontology to the other, and such that the elements of the source ontology appear at most once.

Ontology matching is required in the processes of ontology *integration*, *merging*, and *translation*. Ontology *merging* (Figure 2.3) is the process of generating a single, coherent ontology from two or more existing and different ontologies related to the same subject (Pinto et al. [1999]). In this process, the initial ontologies remain unaltered. The merged ontology is supposed to contain the knowledge of the initial ontologies, e.g., consequences of each ontology are consequences of the merged ontology.

Ontology integration (Figure 2.3) is the inclusion in one ontology of another ontology and assertions expressing the *glue* between these ontologies. The integrated ontology is supposed to contain the knowledge of both initial ontologies. Contrary to merging, the first ontology is unaltered while the second one is modified. According to (Pinto et al. [1999]), ontology integration is the process of generating a single ontology from two or more existing and different ontologies in different subjects.

Ontology translation is the process of transforming an ontology from one ontology language to another.

In this thesis, the focus is on the ontology matching process, which is detailed in the following sections.

2.2.2 The Matching Problem

Ontology is a representation of the reality. The ontology engineering process involves different designers with different views of the reality. The resulting ontologies involves different conventions, granularity, and coverage. Heterogeneity in the applications using these ontologies arise in different levels. The matching problem is related to the heterogeneity problem.

2.2.2.1 Heterogeneity problem

There are different kinds of heterogeneity, which can be grouped into the following categories (Euzenat and Shvaiko [2007]):

- *syntactic heterogeneity*: occurs when two ontologies are not expressed in the same ontology language. For instance, the source ontology is defined in OWL while the target ontology is expressed in RDF.

- *terminological heterogeneity*: occurs due to variations in names when referring to the same entities in different ontologies. For instance, same names describing different concepts or different names describing the same concept (Paper and Article).

- *conceptual heterogeneity* (semantic heterogeneity Euzenat [2001] or logical mismatch Klein [2001]): stands for the differences in modeling the same domain of interest. This happens due to the use of different axioms for defining concepts or due to the use of different concepts. This kind of heterogeneity include (Benerecetti et al. [2001]): *difference in coverage* (two ontologies describe different regions of the world at the same level of detail), *difference in granularity* (two ontologies describe the same region of the world from the same perspective, but at different levels of detail), *difference in perspective* (two ontologies describe the same region of the world, at the same level of detail, but from different perspective – e.g., political and geographic maps).

- *semiotic heterogeneity* (or pragmatic heterogeneity (Bouquet et al. [2004])): is concerned with how entities are interpreted by people. Entities which have exactly the

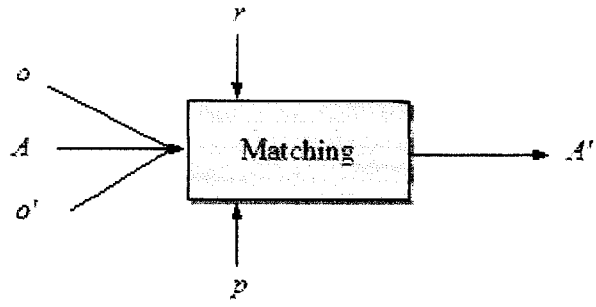

Figure 2.4: The matching process (Shvaiko and Euzenat [2005]).

same semantic interpretation are often interpreted by humans with regard to the context, i.e., how they are ultimately used. This kind of heterogeneity is difficult for the computer to detect and even more difficult to solve (Euzenat and Shvaiko [2007]).

2.2.2.2 The matching process

The *matching* process determines the alignment A' for a pair of ontologies o and o' (Figure 2.4). There are some others parameters that can extend the definition of the matching process (Shvaiko and Euzenat [2005]): (i) the use of an input alignment A, which is to be completed by the process (previous alignment); (ii) the matching parameters, p (weights, thresholds); (iii) external resources used by the matching process, r (common knowledge and domain specific thesauri).

Technically, this process can be defined as follows:

Matching process (Euzenat and Shvaiko [2007]) . The matching process is a function f which has as input a pair of ontologies to match o and o', an alignment A, a set of parameters p, and a set of resources r, and that returns an alignment A' between these ontologies: $A' = f(o,o',A,p,r)$.

2.2.2.3 Structure of an alignment

An alignment is a set of correspondences between entities of different ontologies. A correspondence (or a mapping) considers the two entities and the relation that is supposed to hold between them. A correspondence can be described as a 5-tuple: (id,e,e',n,R):

- id, is a unique identifier of the correspondence;
- e and e' are ontology entities, expressed using the respective ontology language.

- n is a *confidence measure* holding the correspondence between the entities e and e'.
- R is the relation between e and e'.

For instance, the entities e and e' can be expressed in OWL, <owl:Class rdf:ID="Article"> and <owl:Class rdf:ID="Paper">, respectively. The set of relations that can be used for expressing the relations between the entities includes the *equivalence* relation (=), meaning that the matched objects are the same or are equivalent, *disjointness* ($\perp$), indicating that the classes are disjoint, and *more general* ($\sqsupset$), indicating a relation of subsumption between the entities. Classical thesaurus relations (*broadMatch*, *narrowMatch*, and *relatedMatch*), fuzzy relations or probabilistic distributions over a complete set of relations can be also used.

Every relationship between two entities can have a degree of confidence, which can be viewed as a measure of trust in the fact that the correspondences holds. The most widely used structure is based on the real number from the interval [0,1]. However, discrete categories can also to be used, such as proposed in (Trojahn et al. [2008d]), where *certainty* and *uncertainty* values are assigned to the confidences on the correspondence.

2.2.3 Applications

Ontology matching applications range from traditional applications such as ontology engineering and information integration to more recently agent communication, query answering, and navigation on the Semantic Web. In these applications, imposing a central, common ontology is not realistic. So, matching of heterogeneous ontologies is an intrinsic problem. In this section, applications such as ontology engineering, information integration, P2P, agent communication, and navigation on the web, are commented.

Ontology engineering is the task of designing, implementing, and maintaining ontologies. This task involves to deal with multiple and distributed ontologies, which often need to be put together. So, ontology engineering needs support to identify relations between the entities in these ontologies. Moreover, ontology evolution and versioning (Noy and Klein [2004]) is another important task in ontology engineering that requires support of matching.

Information integration is one of the classical matching applications. This task is also referred to in the literature as schema integration (Batini et al. [1986]), data warehousing (Bernstein and Rahm [2000]), data integration (Halevy et al. [2005]), and catalogue integration (Giunchiglia et al. [2005]). The basic idea behind information integration is to provide users with a unified view of different and local information sources, usually on the basis of a global information view over which queries can be expressed. This involves to identify correspondences between semantically related entities in the local sources.

Peer-to-Peer (P2P) is a distributed communication model in which parties (peers) can provide

each other with data and services. Traditional applications describe their data using simple schema, with possible different terminologies to describe the same domain of interest. In order to establish information exchange between peers, the characterization of the relation between their schemas is needed. More recently, semantic P2P (Calvier and Reynaud [2008]) systems have used more complex specification of their contents, using database schemas and formal ontologies.

Web services composition is another ontology matching application. Web services are processes with a well-defined interface, that can be invoked through the Web. Applications use services descriptions in order to discovery and invoke them. A richer and more precise way to describe the services have been proposed using knowledge representation languages and ontologies (Fensel et al. [2007]). Matching operations are required for both tasks (Euzenat and Shvaiko [2007]): (i) compare the descriptions of services, in order to know if they are relevant; and (ii) route the knowledge they process in order to compose different services by routing the output of some service to the input of another service (translation of the output of one service into a suitable input for another service, where for instance, the input and output of the services are described by different ontologies).

Other kinds of application involve distributed and autonomous entities that must communication to each other in order to solve their tasks. Multi-Agent communication is a typical example (see for instance Trojahn et al. [2008b]). Multi-Agent systems are by nature distributed and heterogeneous. In these systems, ontologies play a fundamental role, formalizing the vocabulary from the agent's perception of the world. In open Multi-Agent Systems, such as the Web, agents using different ontologies need to agree on the vocabulary they use, in order to communicate and then resolve their tasks. Ontology matching is a primary problem that has to be solved in order to allow agents with different backgrounds to adjust themselves before starting any form of cooperation or communication. Using a common ontology is unpractical, because it would result in assuming a standard communication vocabulary and it does not take into account the conceptual requirements of agents that could appear in future (Laera et al. [2007]). Moreover, a common ontology forces an agent to abandon its own world view and adopt one that is not specifically designed for its task (van Diggelen et al. [2006]).

The navigation and query answering on the web are other scenarios where matching is required, specially on the semantic web, where ontologies are proposed to be used to describe the content of the available resources. In the navigation scenario, the content of web pages can be annotated with term of ontologies and searching on the web involves matching between the ontologies describing these content (Sabou et al. [2006]). Question answering systems aim to retrieve "answers" to questions rather than full documents or even best-matching passages as most information retrieval systems currently do (Dumais et al. [2002]). Query answering on the web involves two main tasks: rewrite the user query in terms of an ontology and aggregate information derived from multiple heterogeneous data

sources, described by different ontologies.

2.2.4 Ontology Matching Techniques

There are a great variety of ontology matching techniques and strategies proposed in the literature. In this section, the main classifications of these techniques are presented. Next, a new classification is presented. Finally, the basic ontology matching techniques and strategies are commented.

2.2.4.1 Classification of ontology matching techniques

The approaches for ontology matching have been surveyed from different perspectives in (Rahm and Bernstein [2001]), (Wache et al. [2001]), (Kalfoglou and Schorlemmer [2003b]), (Shvaiko [2004]), (Shvaiko and Euzenat [2005]), (Choi et al. [2006]), and (Euzenat and Shvaiko [2007]). These works classify the approaches according to different dimensions. Basically, in (Rahm and Bernstein [2001]), (Shvaiko [2004]), (Shvaiko and Euzenat [2005]) and (Euzenat and Shvaiko [2007]), the approaches are classified in terms of input and techniques utilized in the matching process; (Wache et al. [2001]) classifies the approaches based on the role of the ontology, ontology representation, use of the mappings, and ontology engineering; (Kalfoglou and Schorlemmer [2003b]) present a classification that is based on frameworks, methods and tools, translators, and mediators. (Choi et al. [2006]) present matching classes according to the kind of ontology and operations (i.e., matching between an integrated global ontology and local ontologies, matching between local ontologies, and matching on ontology merging and alignment).

(Euzenat and Shvaiko [2007]) presents general dimensions which are used to classify the matching algorithms:

- input of the algorithms, where the algorithms can be classified depending on the data (instances or schema-level information) or conceptual models used to express the ontologies (OWL or relational models, for instance).
- characteristics of the matching process, which can be based on the nature of its computation (*exact* or *approximate*) and the way the algorithms interpret the input (intrinsic input, external resources, or semantic of the entities).
- output do the algorithms (the form that the results are presented), which concerns the cardinality of the mapping (one-to-one, for instance), the numeric output (confidence degrees, probabilities, or distance measures), and the kind of relation between the entities (equivalence or subsumption, for instance).

In this section, the ontology matching classifications proposed in the literature are grouped into two classes: *standard classifications* and *alternative classifications*. The *standard classifications* include the proposals that are based on similar dimensions, such as kind of input, matching techniques, and output of the algorithms. A more detailed comparison of such classifications can be found in (Trojahn et al. [2005]). The *alternative classifications* use other criteria to classify the matching techniques, such as kind of systems (i.e, frameworks, translations, mediators, etc) and matching categories (i.e., using local or global ontologies), and so on.

Standard classifications

Classification of Rahm and Bernstein [2001]

A classification of ontology matching approaches often used as basis to other works is proposed by (Rahm and Bernstein [2001]). This classification distinguishes between individual and combining matchers (Figure 2.5). Individual matchers compute a mapping based on a single matching criterion, while the combining matchers combine individual matchers in a hybrid matcher (matchers run sequentially and the output of one matcher is used as input to other matcher) or combine multiple match results in a composite matcher (matcher run in parallel and the final results are combined). Basically, the individual matchers comprise:

- Schema-based or instance-based level: match can consider instance data or only schema-level information.
- Element or structure level: match can be performed for individual schema elements (e.g., classes or attributes), or using the ontology structure (i.e., relations between the elements in the ontology hierarchy).
- Linguistic or constrained-based techniques: matcher can use a linguistic based approach (e.g., equality of names, equality of canonical name, synonymous, hypernym, similarity based on common string) or a constraint-based approach (e.g., based on keys, cardinalities, and relationship types).

Classification of Shvaiko [2004]

(Shvaiko [2004]) distinguishes two group of techniques: heuristic or formal techniques; and implicit or explicit techniques (Figure 2.6):

- Heuristic or formal techniques: the characteristic of the heuristic techniques is that they try to find relations which may hold between similar labels or graph structures; while the formal techniques have model-theoretic semantics which is used to justify their results.

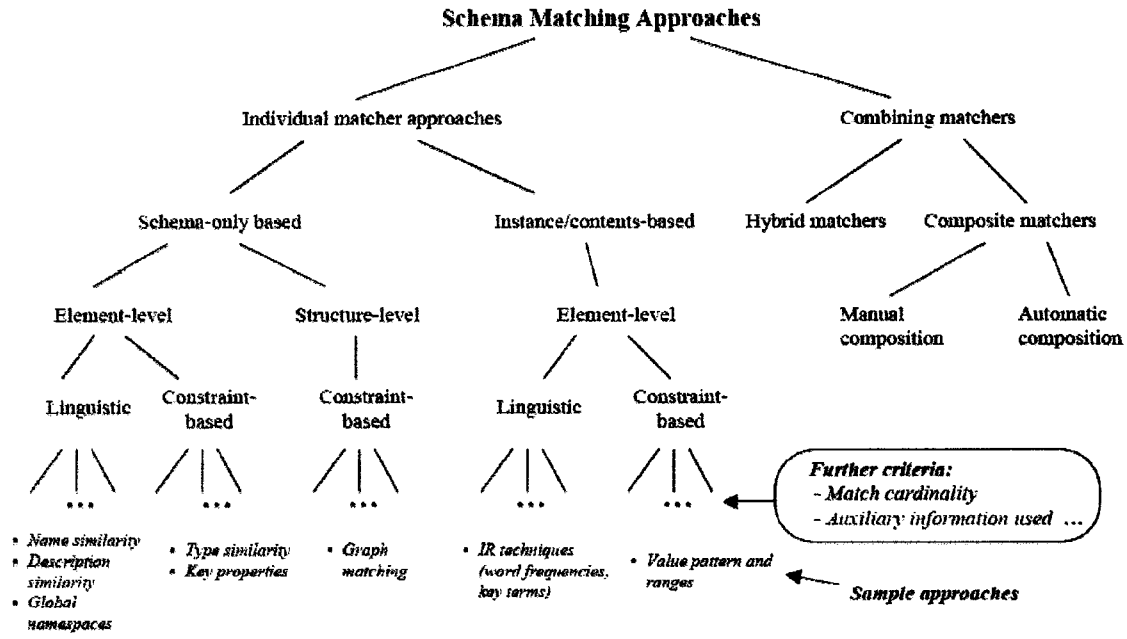

Figure 2.5: Classification of (Rahm and Bernstein [2001]).

- Implicit or explicit techniques: implicit techniques are syntax driven techniques (i.e., techniques which consider labels as strings, analyze data types, or soundex of schema/ ontology elements); explicit techniques exploit the semantics of labels (i.e., techniques are based on the use of tools, which explicitly codify semantic information, such as thesauri).

Classification of Giunchiglia and Shvaiko [2003]

(Giunchiglia and Shvaiko [2003]) classifies the matching approaches into *syntactic* and *semantic*. These categories correspond to syntactic and conceptual categories of (Zanobini [2006]). Basically, the syntactic techniques return coefficients in the [0,1] range, while sematic techniques return logical relations, such as equivalence and subsumption. Moreover, it is distinguished between weak semantics and strong semantics element-level techniques. Weak semantics techniques are syntax-driven techniques (e.g., techniques which consider labels as strings, or analyze data types,) while strong semantics techniques exploit, at the element level, the semantics of labels (e.g., based on the use of thesauri).

Classification of Shvaiko and Euzenat [2005]

Based on the classification proposed by (Rahm and Bernstein [2001]), (Shvaiko and Euzenat [2005], Euzenat and Shvaiko [2007]) introduces new proprieties, distributed in three layers (Figure 2.7):

- *granularity/input interpretation* (Figure 2.7(a)): which is based on (i) the matcher

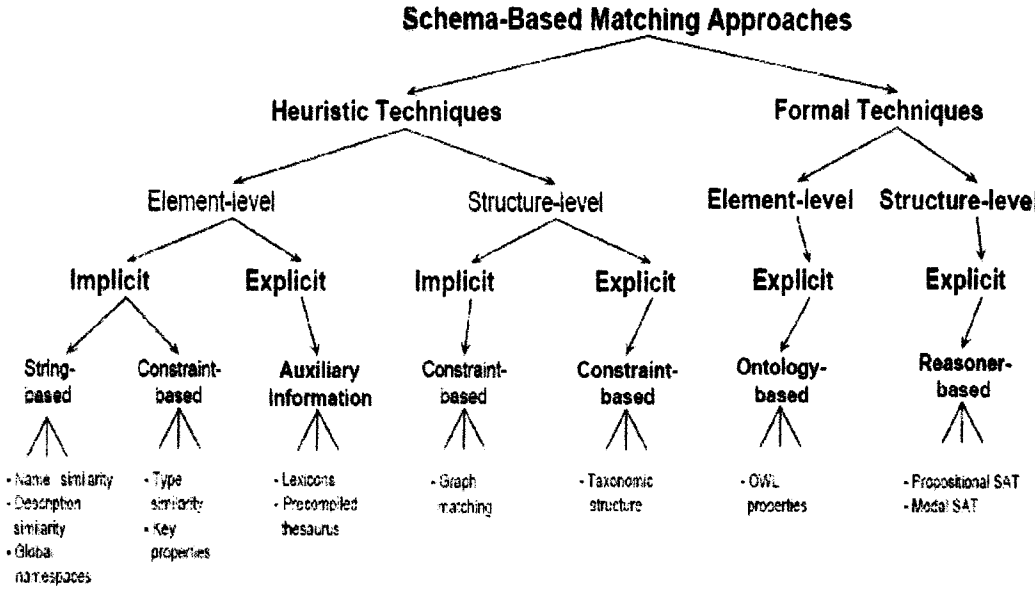

Figure 2.6: Classification of (Shvaiko [2004]).

granularity (element or structure level); and (ii) how the techniques interpret the input information (syntactic, external, or semantic). At *element-level*, the entities are analyzed in isolation, ignoring the relations with other entities; at *structural-level*, the entities are analyzed together the others in the ontology structure. The *syntactic* techniques explore words as a sequence of strings; the *external* techniques explore some external resource to interpret the input, such as thesaurus, upper ontologies, and user input; and the *semantic* methods use formal semantics, such as model-theoretic semantics.

- *basic techniques* (Figure 2.7(b)): which are distinguished by the *granularity/input interpretation*: (i) *element-level*: language-based (which explore morphological proprieties of the input data), string-based (which consider strings as a sequence of letter and are based on the intuition that similar strings denote similar concepts), constraint-based (which use restrictions applied to the definitions of the entities, such as cardinality), linguistic resources (which use lexicons or domain specific linguistic resources), alignment reuse, and upper and domain ontologies (where ontologies are used as external sources of information); (ii) *structural-level*: graph-based (where the input ontologies are viewed as labeled graphs and the similarity comparison is based on the analysis of the positions of terms within the graphs), taxonomy-based (which are similar to the graph techniques, but considering only the specialization relation), repository of structures (which are based on the use of previous calculated similarity between ontologies, not alignments), model-based (which are based on deductive models, such as description logics reasoning techniques), and data-analysis and statistic techniques.

- *kind of input* (Figure 2.7(c)): which is concerned with the type used by a particular technique, depending on the kind of data the algorithms work on: strings (*terminological*), structure (*structure*), models (*semantic*) or data instances (*extensional*). Specifically terminological methods can be string-based (considering terms as a sequence of characters), or based on the interpretation of the terms as linguistic objects (linguistic); and structural methods can consider the internal structure of the entities (internal) (attributes and their types, for instance) or the relations of the entities with other entities (relational).

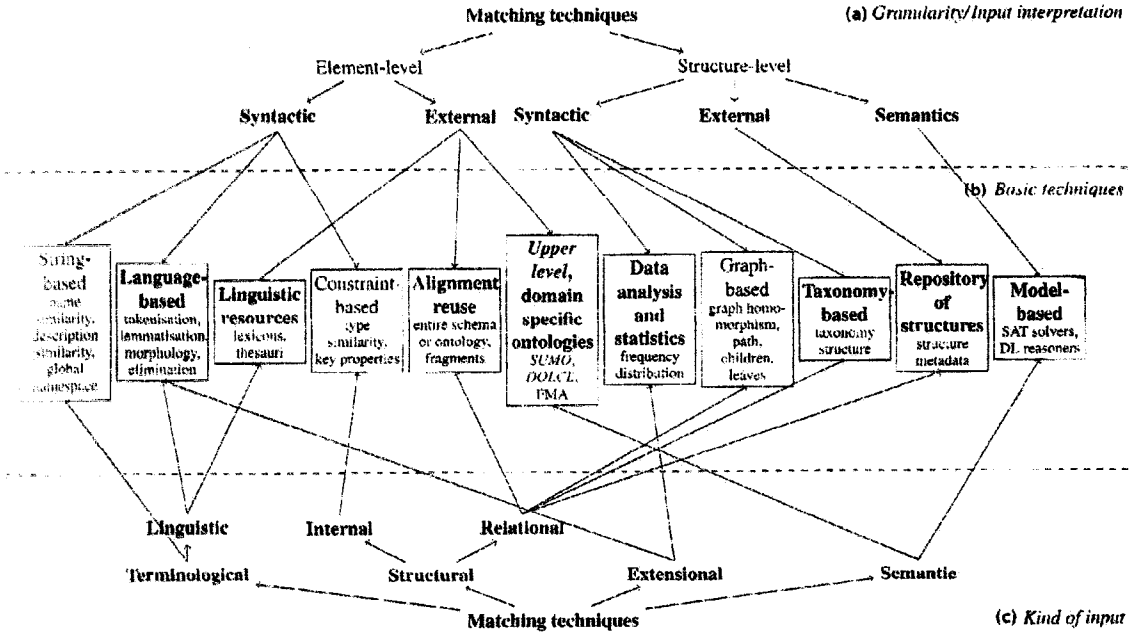

Figure 2.7: Classification of (Euzenat and Shvaiko [2007]).

Alternative classifications

(Wache et al. [2001]) presents an overview of the approaches based on four criteria: (i) use of ontologies, where the role and the architecture of the ontologies influence heavily the representation formalism of an ontology (i.e., if the system use single ontology approaches, multiple ontologies approaches, or hybrid approaches); (ii) ontology representation, where depending on the use of the ontology, the inference capabilities differ from approach to approach (i.e., the system are classified according to the kind of ontology language representation); (iii) use of mappings, which consider if the matching process supports the integration of ontologies (i.e., if the mappings are used to connect sources of information, for instance); and (iv) ontology engineering, which considers if the systems support the reuse or acquisition of ontologies.

(Kalfoglou and Schorlemmer [2003b]) presents a style of ontology matching classification

that is based on: *frameworks* include the systems that provide a combination of tools and a methodological approach to matching; *methods and tools* include either stand-alone or embedded in ontology development environments, and methods used in ontology matching; *translators* include the systems that are used at the early phases of ontology matching; *mediators*, which provide insights on algorithmic issues for mapping programs; *techniques*, which are similar to methods and tools, but not so elaborated or directly connected with matching; and *theoretical frameworks*, which include the theoretical work has not been exploited yet by ontology mapping practitioners.

(Halevy et al. [2005]) classifies matching techniques into (i) rule-based and (ii) learning-based. Rule-based techniques use schema-level information, such as entity names and structures. An example of matching rule is if two entities have similar names or if they have the same number of neighbor entities, then they must be matched. Learning-based approaches use instance-level information, comparing, for instance, the distribution of data instances.

(Zanobini [2006]) classifies the matching methods into three categories, following the cognitive theory of meaning and communication between agents:

- *syntactic*, which is based on purely syntactic matching techniques, such as edit distance between strings and tree edit distance.
- *pragmatic*, which represents methods based on comparison of data instances, including automatic classifiers and formal concept analysis.
- *conceptual*, which works with concepts and compare their meaning, such as exploiting external resources as WordNet to compare senses among the concepts.

The vertical dimension represents specific *domain knowledge*, which can be situated at any layer of the horizontal dimension.

(Do [2006]) extends the work of (Rahm and Bernstein [2001]) by adding a *reuse-oriented* category of techniques on top of schema-based vs. instance-based separation, meaning that reuse-oriented techniques can be applied both at schema and instance levels.

(Choi et al. [2006]) present the broad scope of ontology matching, matching categories (matching between an integrated global ontology and local ontologies, mapping between local ontologies, and mapping on ontology merging and alignment), their characteristics, and a comprehensive overview of ontology matching tools and systems.

(Ehrig [2007]) introduces a classification based on two dimensions: horizontal and vertical. The horizontal dimension includes three layers that are built one on top of another:

- *data layer*, where the matching between the entities is performed using only data values.

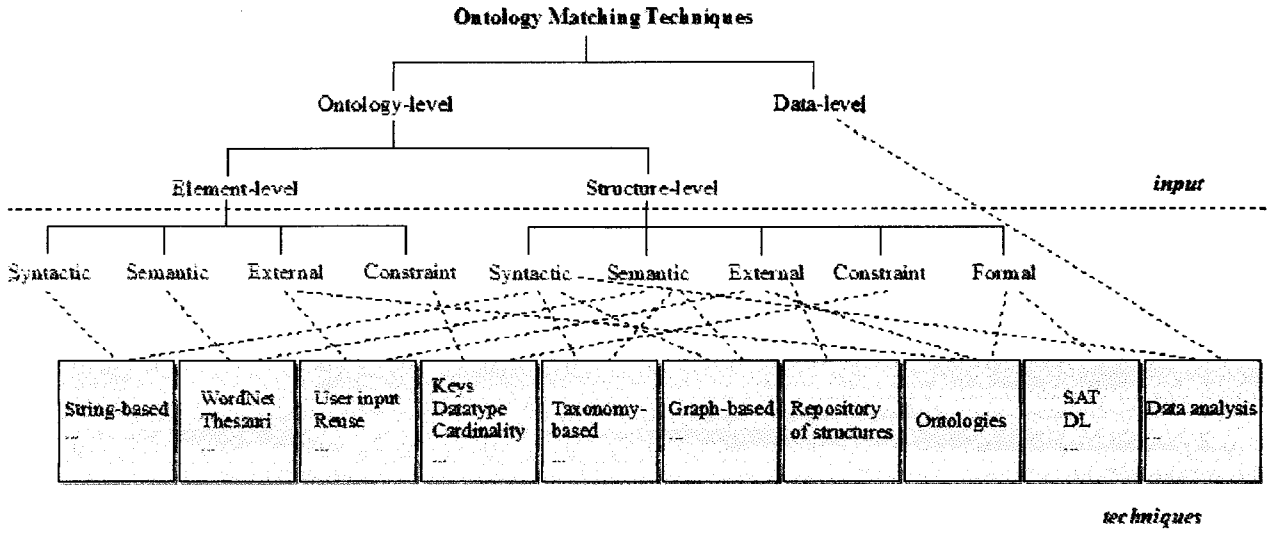

Figure 2.8: Proposed classification of matching approaches.

- *ontology layer*, which is divided into four levels: semantic nets (where ontologies are viewed as graphs with concepts and relations, and matching is performed by comparing only these), description logics (where matching is performed determining taxonomic similarity based on the number of subsumption relations), restrictions, and rules (for these two last levels, the matching is performed considering that if entities have similar constraints, they are similar).

- *context layer*, which is concerned with the usage of entities in the context of an application, and the matching is performed comparing the usages of the entities (i.e., similar entities are used in similar contexts).

Proposed classification

Based on the classifications of (Rahm and Bernstein [2001]) and (Euzenat and Shvaiko [2007]), a new classification of ontology matching approaches is proposed, which is shown in Figure 2.8. Such classification represents an alternative view on the existing matching approaches. It is inspirited on previous specification of different group of matchers, as proposed in (Trojahn et al.). A previous version of the proposed classification can be found in (Trojahn et al. [2008d]).

The classification presents two layers: (i) kind of input, and (ii) basic techniques. Regarding the kind of input, two top input levels are considered: *ontology-level* and *data-level*, which are inspired on the classification of (Ehrig [2007]). At this level, differently from (Rahm and Bernstein [2001]), it is considered only the individual matching approaches. It is assumed that the *combining matchers* are group of matching strategies, which are commented in the next sections.

At the *ontology level*, as well considered by (Rahm and Bernstein [2001]), the approaches can be based on the use of terms of the ontology, without considering their relations with the other terms (*element-level*), or they can take into account the ontology hierarchy and the relations between terms that are neighbors (*structure-level*). On the other hand, at the *data level*, the instances of the ontologies in order are used to compute the alignment.

Based on the kind of input, different basic techniques can be distinguished. Both approaches using element or structural ontology objects basically apply *syntactic*, *semantic*, *external*, or *constraint* techniques. *Formal* techniques are used in general when considering the ontology hierarchy.

Broadly speaking, the *syntactic* approaches use metrics to compare string similarity. This kind of approach is called *linguistic* according to (Rahm and Bernstein [2001]). The *semantic* approaches consider semantic relations between concepts to measure the similarity between them, usually on the basis of one thesaurus or similar semantic oriented linguistic resources. Such approaches are considered as *linguistic* by (Euzenat and Shvaiko [2007]). The *external* matchers consider some type of external information, such as user input or previous matching results. Finally, the *constraint-based* matchers are based on the analysis of data types, value ranges, uniqueness, cardinalities, and other information constraints. For example, the similarity between two terms can be based on the equivalence of data types and domains, of key characteristics (e.g., unique, primary, foreign), or relationship cardinality (e.g., 1:1 relationships) (Rahm and Bernstein [2001]).

It is not included in the proposed classification the *language-based* approaches, as presented by (Euzenat and Shvaiko [2007]). This kind of approach includes techniques such as tokenization (which consists of segmenting strings into sequences of tokens by a tokenizer), lemmatization (which reduces the tokens to a normalized basic form, suppressing tense, gender, and number), and elimination of stop-words (where articles, prepositions, conjunctions are removed, considering that they are non meaningful words). In the context of this thesis, such techniques are considered earlier steps of the matching process.

In the next section, the techniques presented in the second layer of the proposed classification are detailed.

2.2.4.2 Basic techniques

Syntactic techniques

The syntactic techniques are based on the similarity between strings (i.e., the objects of the ontologies are consider as a sequence of characters). This kind of technique is used both at element and structural levels. Such techniques (string-based) include *string equality*, *substring* or *subsequence* techniques, *edit distance*, *statistical measures*, and *path comparison*

(specially used at structural level).

The *string equality* returns 0 if the strings are not identical and 1 if they are identical. This measure does not indicate how different the strings are. An alternative measure used is the Hamming distance, which counts the number of positions in which the strings differ, where s is the term of the source ontology and t is the term of the target ontology:

$$\delta(s,t) = \frac{(\sum s[i] \neq t[i]) + \mid\mid s \mid - \mid t \mid\mid}{max(\mid s \mid, \mid t \mid)}$$

A single *substring* comparison verifies if the one string is substring of another, returning 1 if yes or 0 if not. A more refined measure computes the ratio of the common subpart between two strings (where x is the longest common substring of s and t):

$$\sigma(s,t) = \frac{(2 \mid x \mid)}{\mid s \mid + \mid t \mid}$$

Variations of the substring measures can consider prefix and suffix substrings. The *n*-gram similarity is also used to compare strings. It computes the number of common sequence of n characters between the strings. For instance, trigrams for the string article are: art, rti, tic, icl, cle.

$$\sigma(s,t) = (\mid ngram(s,n) \cap ngram(t,n))$$

An *edit distance* between two strings is the minimal cost of operations to transform one string into another. One well-known measure is the Levenshtein distance or edit distance (Levenshtein [1966]), which is given by the minimum number of insertions, deletions, and substitutions of characters needed to transform one string into another. For example, for the words "score" and "store" the Levenshtein metric returns 0.68.

Based on Levenshtein measure, (Maedche and Staab [2002]) proposes a string similarity measure for strings, the String Matching (SM), that considers the number of changes that must be made to change one string into the other and weighs the number of these changes against the length of the shortest string of these two, where *ED* is the edit distance between s and t:

$$SM(s,t) = max\left(0, \frac{min(\mid s \mid, \mid t \mid) - ED(s,t))}{min(\mid s \mid, \mid t \mid)}\right)$$

Statistical methods are based on the intuition of identifying strings as *bags of words*, in which information retrieval techniques can be applied. Such techniques are based on the corpus of

such strings, i.e., the set of all such strings found in one of the ontologies or in both of them. A common measure is TFIDF (Term frequency-inverse document frequency, which is used for scoring the relevance of a bag of words (document) to a term by taking into account the frequency of appearance of the terms in the corpus.

Specially at structural level, *path comparison* methods are applied. It uses the sequence of labels of entities to which those bearing the label are related. A usual practice is to concatenates all the names of super-classes of the classes before applying a string based approach. This kind of approach includes both *graph-based* (where the similarity comparison is based on the analysis of the positions of terms within the graphs), and *taxonomy-based* (which are similar to the graph techniques, but considering only the specialization relation).

An ontology can be considered a graph whose edges are labeled by relations. Basically, the relational structures techniques consider three kinds of relations: involves taxonomic structures, mereologic relations, and all the involved relations. Considering the taxonomic structure, measures to compare the classes based on relations of subclasses are applied. A common method is to count the number of edges in the taxonomy between two classes. The structural topological similarity on a hierarchy follows the graph distance. Other techniques based on the taxonomic structure consider: (i) super and subclasses rules, which capture the intuition that classes are similar is their super or subclasses are similar; (ii) bounded path matching, which take two paths with links between classes defined by the hierarchical relations, compare terms and their positions along this paths, and identify similar terms.

The techniques based on the mereologic structure, which corresponds to the *part-of* relationship, use the following intuition: classes are similar if their parts are similar.

Semantic techniques

The problem in comparing ontology entities on the basis of the labels as a sequence of strings occurs due to the existence of synonymous (different words used to name the same entity) and homonymous (are words used to name different entities). For example, for the above example, the words "score" and "store", which have as Levenshtein similarity 0.68, represent very different concepts. On the other hand terms like "student" and "learner" are semantically similar although they are syntactic distant from each other. It is not always correct to deduce that two entities are the same if they have the same name or that they are different because they have different names, due to syntactic variations of the same word that occurs according to different acceptable spellings, abbreviations, use of optional prefixes and suffixes (compact disc vs. CD, CD-ROM).

The semantic approaches use external resources, such as: (i) lexicons, a set of words with a natural language definition of them; (ii) thesauri, a kind of lexicon, containing relations such as synonym and hypernym; and (iii) terminologies, which contains phrases rather than words. These resources are often domain specific. A well-know thesauri is the WordNet, an electronic lexical database for English. WordNet is based on the notions of synsets, which

denote a concept or a sense of a group of terms. Other relations are provided by the WordNet, such as hypernym and hyponym (superclasse and subclasse, respectively), and meronym (part of relation).

Using WordNet to ontology matching, three families of methods can identified (Euzenat and Shvaiko [2007]): (i) terms are similar because they belong to the same synset; (ii) using the hypernym structure to measure the distance between synsets corresponding to the terms; (iii) using the definitions of the concepts to measure the distance between the synsets of the terms. The relations provided by WordNet can be used to specify the logical relations between the terms (Giunchiglia et al. [2004b]): equivalence (terms are synonymous), subsumption (terms are related by holonym or hypernym), disjunction (terms are antonyms). Moreover, the match can be based on the comparison of the glosses of the terms, verifying the number of co-occurrences of labels in the glosses of the terms (i.e., if the number is greater than a threshold, the terms can be considered similar).

The semantic relations provided by the thesaurus can be used when considering the structure of the ontologies (graph-based and taxonomy-based matchers).

External-based information techniques

This kind of technique is based on the use of external information, such as user input or iterative feedback, reuse of alignment, repository of structures, and ontologies. Reuse of alignments techniques are based on the use of alignment of previously matched ontologies. The idea behind this kind of technique is that many ontologies to be matched are similar to already matched ontologies, specially if they are describing the same application domain (Euzenat and Shvaiko [2007]). The proposals for reuse have been presented in (Rahm and Bernstein [2001]). Upper level and domain ontologies are seen as external sources of common knowledge, which can provide provide formal specifications that can be explored in the matching process. For instance, domain ontologies can provide structural information for matching process dealing with simple structured ontologies. Repository of structures store ontologies and their fragments together with pairwise similarity measures. Unlike alignment reuse, repositories store only similarity between ontologies, not alignments (Euzenat and Shvaiko [2007]). In the matching process, the structures to be matched are checked against the stored structures, in order to avoid the matching operation over dissimilar structures.

Constraint-based techniques

Constraint approaches compute the similarity between the entities based on set of proprieties, their cardinality, and the transitivity or symmetry of their properties. Considering that correspondences between ranges and domain of entities tend to be enormous, these kinds of methods are more likely to be used to create correspondences clusters rather than to discover accurate correspondences between the entities. They are used in combination with other element-level.

A common method, specially used in database integration, is to compare the proprieties that are key. Using this restriction together with the class similarity, these proprieties can be matched. Other kind of proprieties comparison considers the datatypes (range) and domain. While the datatypes indicate the way the the values are stored in a computer (e.g., integer, float, etc), the domain and range indicate the classes the properties are associated with.

Formal techniques

The main characteristic of the formal techniques is that model-theoretic semantics is used to justify their results (Euzenat and Shvaiko [2007]). The basis of this techniques are to merge the two ontologies and to search for correspondences A, such that $o,o' \models A$.

Commonly these methods are used together with other ones, typically those that provides an initial set of equivalent entities. This group of approaches include: (i) techniques based on external ontologies; and (ii) deductive techniques. In the first group, external ontologies can be used as a kind of *background knowledge*. The basic intuition is that a background ontology, with a coverage of the domain of interest of the ontologies to be matched, helps in the disambiguation of multiple possible meanings of terms. The alignments found using these methods can be explored by deductive techniques. The deductive techniques include propositional and Description Logics techniques.

An approach for applying propositional satisfiability (SAT) techniques includes the following steps (Euzenat and Shvaiko [2007]):

1. Build a theory or domain knowledge (*axioms*) for the given input two ontologies as a conjunction of the available axioms. The theory is constructed by alignments provided by matchers.

2. Build a matching formula for each pair of classes c and c' from two ontologies. For each pair c, and c' for which is needed to test the relation r (i.e., $=$, $\sqsubseteq$, $\sqsupseteq$, $\perp$), a matching query is created: $Axioms \rightarrow r(c,c')$

3. Check for validity of the formula (i.e., that it is true for all truth assignments of all the propositional variables occurring in it.

Using Description logics techniques, the relations ($=$, $\sqsubseteq$, $\sqsupseteq$, $\perp$) are expressed using subsumption. Merging two ontologies (after renaming) and testing each pair of concepts and roles for subsumption is sufficient to match terms with the same interpretation (Bouquet et al. [2006]).

Data-level techniques

The data-level techniques (or extensional techniques as refereed by Euzenat and Shvaiko [2007]) are based on the instances comparison. According to (Euzenat and Shvaiko [2007]),

the extensional methods can be divided into three categories: (i) common-extension techniques, which use common instance sets of the ontologies; (ii) instance identification techniques; and (iii) disjoint extension comparison, which work on heterogeneous sets of instances.

In the first group of techniques, the comparison between classes involves to verify the intersection of their set of instances set A and B. If $A \cap B = A = B$, the classes can be considered very similar, and more general if $A \cap B = A$ or $A \cap B = B$. A well-known measure used to compute the distance between two sets of instances is Hamming distance, which corresponds to the size of the symmetric difference normalized by the size of the union:

$$\sigma(x, y) = \frac{|\, x \cup y - x \cap y \,|}{|\, x \cup y \,|)}$$

In order to compute the similarity based on the probabilistic interpretation of the set of instances, the Jaccard similarity is adopted (where P(X) is the probability of a random instance to be in the set X):

$$\sigma(A, B) = \frac{P(A \cap B)}{P(A \cup B)}$$

The instance identification techniques are commonly used when the common set of instances is not available. The aim of these techniques is to identify the instances from one set that corresponds to the another set. A general method in this category comprises the comparison of values of the proprieties.

In the cases where it is not possible to infer a common set of instances, disjoint extension comparison can be used. These methods can be based on statistical measures about the features of class members, on the similarities computed between instances of classes, or based on a matching between entity sets (Euzenat and Shvaiko [2007]). The statistical about the property values of the instances involves measures such as maximum, minimum, mean, variance, which should be the same for equivalent classes of different ontologies, when there are statistically representative examples. Similarity-based techniques typically compute the distance between the set of instances, using similarity measures between the instances which can be done using other basic methods (this kind of approach differs from the common extension approach, which always returns 0 if the two classes do not share any instance). The average linkage can be used to compute the distance between all the possible comparisons between the instances. Finally, matching-based comparison consider that the elements to be compared are the most similar. The distance between two sets is a value to be minimized and its computation is a optimization problem: to find the elements of both sets that correspond to each other.

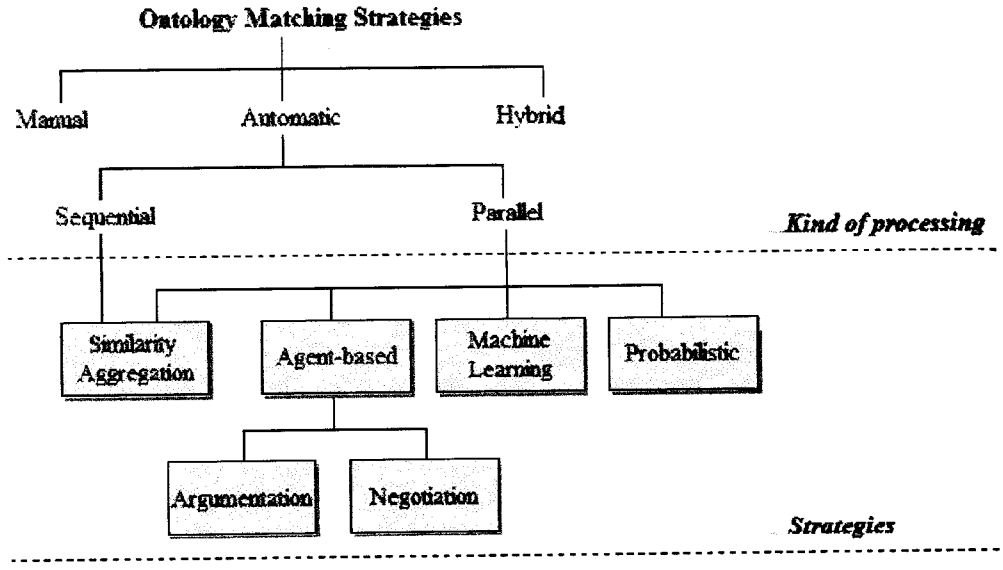

Figure 2.9: Matching strategies.

2.2.4.3 Matching strategies

The matching problem is not limited to the use of techniques individually. The matching strategies aim to combine different techniques. They range from the combination of techniques using aggregated similarity measures to methods that are used to learn the best technique. Figure 2.9 shows a schematic classification of the matching strategies, which is composed by two layers: (i) *kind of processing* and (ii) *basic strategies*. At the first layer, the strategies can be classified into three groups: *manual*, *automatic*, and *hybrid*.

The *automatic* processing involves *sequential* or *parallel* processing. A basic way to combine matchers comprises the sequential composition, where the output of a matcher is used as output for the other matcher. This kind of processing is considered *hybrid* in the classification of (Rahm and Bernstein [2001]). Another way to combine matchers consists of running several matcher independently and aggregating their results – parallel composition according to (Euzenat and Shvaiko [2007]) or composite following (Rahm and Bernstein [2001]).

According to the kind of processing, different strategies can be used to combine matchers results. Such strategies can be classified in *similarity aggregation*, *agent-based*, *machine learning*, and *probabilistic methods*. The similarity aggregation involves computing the similarity between two entities as aggregation of multiple similarities. For instance, the similarity between two classes can be computed by aggregating, in a single similarity measure, the similarity obtained from their names and the similarity of their superclasses (Madhavan et al. [2001]). It can include weighted sum and weighted average of the similarities. Moreover, aggregation can correspond to choosing one of the results on some criterion or merging their

results through some operator (intersection or union, for instance).

Another way to combine matcher results is using a agent-based approach, where agents encapsulate different matching algorithms and use approaches such as negotiation and argumentation to combine the individual results.

The learning methods learn how to sort alignments through the presentation of correct (positive examples) and incorrect alignments (negative examples). Matchers using machine learning operate in two steps (Euzenat and Shvaiko [2007]): (i) the learning or training phase; and (ii) the classification or matching phase. In the first step, training data for the learning process is created, for instance, by manually matching two ontologies, and the system learns a matcher from this data. In the second step, the learnt matcher is used for matching new ontologies. The well-known machine learning methods, such as Bayes learning, neural networks, and decision tree have been applied to ontology matching.

Similarly to learning methods, probabilistic methods, such as Bayesian networks, can be used to match ontologies, in order to enhance matching candidates or combine matchers.

2.3 Summary

In this chapter, an introduction about ontologies and a review on ontology matching techniques and strategies were presented. A proposal of classification of matching approaches was presented.

The matching techniques can be grouped into broad categories: lexical (detecting similarities between labels of concepts), semantic (the terms can be evaluated semantically, usually on the basis of semantic-oriented linguistic resources), structural (using the structure of the ontologies), and instance-based mapping (using instance data to detect the similarity between concepts). However, each category offers a wide diversity of options. An important aspect of ontology matching is to find ways of choosing among these techniques and their variations and then combining their results. In the context of this thesis, strategies based on agent approach, such as negotiation and argumentation, are proposed. A review of these techniques is presented in the next chapter.

Chapter 3

Agent Cooperation

Researches in Multi-agent Systems (MAS) are motivated by the conceptualization, design, and implementation of entities (agents) that operate in a distributed and open environment. The key characteristics of MAS are (Sycara [1998]) that (i) each agent has incomplete information or capabilities for solving the problem and, thus, has a limited viewpoint; (ii) there is no system global control; (iii) data is decentralized; and (iv) computation is asynchronous.

In order to solve conflicts that arise when several agents need to cooperate and coordinate their activities, mechanisms of negotiation and argumentation are used. Both approaches refer to some kind of communication between agents. However, systems involving negotiation and argumentation are very different in their nature ranging from communication in form of auctions to argumentation in a more philosophical sense.

Negotiation involves the communication between agents that iteratively exchange proposals and counterproposals. Argumentation can be seen as a more sophisticated exchange of deals in a negotiation protocol or a model for reasoning based on the construction and comparison of arguments. In the first sense, arguments are seen as meta-information used to justify the negotiation proposals or to persuade agents to change their negotiation stance. In the second sense, argumentation can be abstractly defined as the interaction of different arguments for and against some conclusion.

In this chapter, negotiation and argumentation are presented, where argumentation is both considered as a sophisticated sub-class of the negotiation approaches and a reasoning model to deal with different levels of the comparison of arguments. The chapter is organized as follows. Section 3.1 comments on the negotiation approach. Basically, the negotiation process, negotiation models and techniques are presented. Section 3.2 presents the argumentation approach, considering the argumentation frameworks proposed in the literature. Finally, Section 3.3 presents the summary of the chapter in the context of the thesis.

3.1 Negotiation

Negotiation is the process by which two or more parties make a joint decision (Zhang et al. [2005]). It is a key form of interaction that enables groups of agents to arrive at mutual agreement regarding beliefs, goals or plans (Beer et al. [1999]). Hence the basic idea behind negotiation is reaching a consensus (Green et al. [1997]), negotiation can be any communicative process that results in mutually acceptable agreements. As (Jennings et al. [2001]), the minimum requirement of a negotiating agent is the ability to make and respond to proposals, where a proposal is a solution to the negotiation problem. In its broadest sense, negotiation involves the design of high-level protocols for agent interaction (Lomuscio et al. [2001]). The term is present in many different fields and, as a result, several definitions have been proposed in the literature. Basically, it refers to communication processes that support cooperation and coordination in MAS.

Negotiation has its origin in both Distributed Problem Solving (DPS), where the agents are assumed to be cooperative, and Multi-agent Systems (MAS), where the agents are supposed to be self-interest. However, there are also proposals in MAS on mechanisms for cooperative agents who need to resolve conflicts that arise from conflicting beliefs about different aspects of their environment. In DPS, negotiation is used for distributed planning and distributed search for possible solutions for hard problems. In MAS, negotiation is used to find ways in which agents can negotiate to come to an agreement, giving the agents the means to resolve their conflicting objectives, correct inconsistencies in their knowledge of other agents' world view, and coordinate a joint approach to domain tasks which benefits all the agents concerned (Parsons and Jennings [1996a]).

Depending of the degree of cooperation between the agents, negotiation can be cooperative or competitive (Guttman and Maes [1998]). Competitive negotiation can be described as a process of making decision to solve conflicts involving goals mutually exclusive. The agents involved in this kind of negotiation are self-motivated and seek to maximize their individual utility. Competitive negotiation is commonly adopted in scenarios based on the game theory (Zlotkin and Rosenchein [1996]) and electronic commerce (Fatima et al. [2004], Sandholm [2000]).

On the other hand, cooperative negotiation is the process of making decision to solve conflicts involving multiples and interdependent goals, not mutually exclusive. In this kind of negotiation, the agents are cooperative and collaborate to seek a common goal, according to the system's global goals (agents work towards satisfying the same goal). In cooperative negotiation, each agent has a partial view of the problem and the results are put together via negotiation trying to solve the conflicts posed by having only partial views (Gatti and Amigoni [2004]). Commonly, cooperative negotiation is applied in scenarios involving resource and task allocation (Bigham and Du [2003], Mailler et al. [2003], Zhang et al. [2005]).

In the next section, the negotiation process and its proprieties are commented.

3.1.1 Negotiation Process

Basically, the negotiation process is based on the exchange of proposals, critiques, counter-proposals, and explanations (Parsons and Jennings [1996b], Parsons et al. [1998]). It usually proceeds in a series of rounds, with every agent making a proposal - a kind of solution to the problem - at each round. One agent generates a proposal and the other agents review it. If some other agent does not like the proposal, it rejects the proposal and generate a kind of feedback in the form either of counter-proposal (an alternative proposal generated in response to the initial proposal) or critique (comments on which parts of the proposal the agents likes or dislikes). The other agents (including the agent that generated the first proposal) review the feedback. From such feedback, the proposer should be in a position to generate a proposal that is more likely to lead to an agreement. In addition to generating proposals, counter-proposals, and critiques, the agents can make the proposal more attractive by providing additional meta-level information in the form of arguments for its position. The process is then repeated. It is assumed that a proposal becomes a solution when it is accepted by all agents.

3.1.1.1 Negotiation Process Proprieties

(Lomuscio et al. [2001]) groups the parameters on which the negotiation can take place into the following broad proprieties:

- *cardinality of the negotiation*: which can be distinguished between the cardinalities of the negotiation domains itself (single-issue or multiple-issue) and of the interaction that take place between the agents (one-to-one,many-to-one, many-to-many). Single-issue negotiation involves only one issue, such as price. In case of multiple-issue negotiation, different issues can be related by some utility function. The interactions between agents can be classified in terms of the number of agents participating in the negotiation: one-to-one (where one agent is negotiating with exactly one other agent), many-to-one (where many agents negotiate with just one agent), and many-to-many (where many agents negotiate with many other agents).

- *agent characteristics*: each agent is assumed to be capable of rating its preferences, so that it can evaluate and choose between different deals. The agents are characterized depending on their: role (i.e., buyer or seller), rationality, knowledge, commitment, social behavior, and bidding strategy.

- *environment* and *goods characteristics*: the negotiation environment can be static or dynamic, depending if the variables are constant or change over time, respectively. The goods can be private or public: agents can value the goods differently depending on whether it is intended for private use or whether its value depends on how other agents value it.

- *event parameters*: offers and other events have a set of proprieties, which form a important part of the specification of the protocol - bid validity (to be valid, bids have to be offered at an appropriate time and must satisfy some constraints on their value), bid visibility (private or broadcast), and clearing schedule and timeouts.

3.1.2 Negotiation Model

A negotiation model is composed by a *negotiation protocol* and *negotiation strategies* (Lomuscio et al. [2001], Wooldridge [2002]). The *negotiation protocol* defines the set of rules that govern the interaction. This covers the permissible types of participants (e.g., the negotiators and relevant third parties), the negotiation states (e.g., accepting bids, negotiation closed), the events that cause state transitions (e.g., no more bidders, bid accepted), and the valid actions of the participants in particular states (e.g., which can be sent by whom, to whom and at when). The protocol defines the circumstances under which the interaction between the agents takes palace: what deals can be made and what sequence of offers are allowed (Jennings et al. [2001]).

The *negotiation strategy* is the specification of the sequences of actions (usually proposals and counter-proposals) the agent plans to make during the negotiation. A strategy is the way an agent behaves in an interaction. The exact deals that an agent proposes is a result of the strategy which this agent uses. For instance, an agent could concede at the first round or bargain very hard throughout the negotiation until a private timeout is reached. Analogously, while a protocol describe the rules governing movement of pieces in the game, a strategy is the way in which a player decides on his next move.

The negotiation strategy an agent employs is critical with respect to the outcome of the negotiation. The strategies that perform well with certain protocols can not perform well with others: the choice of strategy to use is thus a function not just specific of the negotiation scenario, but also of the protocol in use (Lomuscio et al. [2001]).

According to (Jennings et al. [2001]), a negotiation model is formed by three elements: *negotiation protocol*, *negotiation objects*, and *agent decision making models*. The *negotiation objects* cover the range of issues over which agreement must be reached. The object may contain a single issue (price), or it may cover multiple issues (price, quality, penalties, etc). Moreover, agents can have the flexibility to change the values of the issues in the negotiation object and dynamically to alter the structure of the negotiation object (by adding or removing

issues). Such objects characteristics correspond to the properties of the negotiation models, as described in Section 3.1.1.1.

The *agent's decision making models* cover the decision making apparatus the participants employ to act in line with the negotiation protocol, in order to achieve their objectives. The sophistication of the model, as well as the range of decisions that have to be made, are influenced by the protocol in place, by the nature of the negotiation object, and the range of operation that can be performed on it. The decision making mechanisms describe the possible set of agent strategies in using the protocol (Lomuscio et al. [2001]).

In some works, the negotiation protocol is the dominant concern (Rosenschein and Zlotkin [1994], Sabater et al. [2000]), focusing on the types of operation that can be performed on the negotiation object and the behavior of the agents' decision making models. In other cases, the agent's decision making model is the dominant concern (Sierra et al. [1999]), where the protocol does not prescribe an optimal strategy for the agent and there is scope for strategic reasoning to determine the best course of action. In such cases, the relative success of two agents is determined by the effectiveness of their reasoning model. In the next section, the main negotiation protocols are presented.

3.1.2.1 Negotiation Protocols

Negotiation protocols may be of different forms. Depending on the protocol type, a negotiation can be categorized as an auction, a contract-net protocol, or a voting or bargaining scheme (Winoto et al. [2002]). The protocols can be evaluated according to many criteria, and the choice of the protocols depends on what properties the protocol designer wants the overall system to have. (Sandholm [1999]) proposes a group of evaluation criteria: (i) *social welfare*, which is the sum of all agents' payoffs or utilities in a given solutions and it can be used to measure the global goods of the agents; (ii) *Pareto efficiency*, which is used to evaluate solutions, i.e., a solution x is Pareto efficient if there is no other solution x' such that at least one agent in better off in x' and no agent is worse off in x' than in x; (iii) *individual rationality*, where a participation in a negotiation is individually rational to an agent if the agents's payoff in the negotiation solution is no less than the payoff that the agent would get by not participating in the negotiation; (iv) *stability*, which indicates that the agent behave in the desired manner (e.g., following the well-known *Nash equilibrium*, each agent chooses a strategy that is a best response to the other agents' strategies); (v) *computational efficiency*; and (vi) *distribution and communication efficiency*, where is desirable to minimize the amount of communication that is required to converge on a good global solution.

This section describe the main kinds of protocols described in the literature.

Protocol based on voting

In a voting setting, all agents give input to a mechanism and the outcome that the mechanism chooses based on these inputs is a solution for all of the agents (Sandholm [1999]) The classical goal of the voting has been to derive a social choice rule that ranks feasible social outcomes based on individual's rankings of those outcomes. Examples of protocols based on voting include: the *plurality protocol*, which is a majority voting protocol where all alternatives are compared simultaneously and the one with the highest number of votes wins; the *binary protocol*, where the alternatives are voted on pairwise, and the winner stays to challenge further alternatives while the loser is eliminated; and the *Borda protocol*[1], which can be described as a mechanism that defines in principle, that points are allocated to alternative strategies. For instance, in a set of X alternatives X points will be allocated to the most preferred strategy, X-1 to the next best, and so on down to the least preferred strategy, which is allocated one point. The protocol requires that all voters have to rank their preferences among the X alternatives. The preferences are collected centrally to rank the scores given to each strategy, and to select the strategy with the maximum score as the winner. The Borda Count mechanism is identified as the unique voting method to represent the true wishes of the voters.

Auctions

Unlike voting, where the outcome binds all agents, in auctions the outcome is usually a deal between two agent roles: auctioneer and bidder. While in voting it is assumed that the social good is enhanced, in auctions, the auctioneer wants to maximize his own profit. Auctions are usually applied on situations where the auctioneer wants to sell an item and get the highest possible payment for it while the bidders want to acquire the item at the lowest possible price.

There are two patterns of interaction in auctions: *one-to-many auction protocols*, where one agent initiates an auction and a number of other agents can bid in the auction; and *many-to-many*, where several agents initiate an auction and several other agents can bid in the auction. Examples of auction protocols include the *English* auction, where each bidder is free to raise his bid and when no bidder is willing to raise anymore, the auction ends, and the highest bidder wins the item at the price of his bid; the *first-price sealed-bid* auction, where each bidder submits one bid without knowing the others' bids, and the highest bidder wins the item and pays the amount of his bid; the *second-price sealed-bid* (Vickrey) auction, where each bidder submits one bid without knowing the other's bid and the highest bidder wins, but at the price of the second highest bid; and the *Dutch* auction, where the seller continuously lowers the price until one of the bidders takes the item at the current price.

Coalition formation

Coalition formation methods allow agents to join together and are thus necessary in cases where tasks can only be performed cooperatively by groups (Kraus et al. [2003]). By

[1]http://www.fipa.org/docs/input/f-in-00089/f-in-00089.htm

creating coalitions, agents can share resources and cooperate on task execution, increasing their benefits. Cooperative game-theoretic models can be used to do this for self-motivated agents each of which has tasks it must fulfil and resources it needs to complete these tasks. Although the agents can act and reach goals by themselves it may be advantageous to join together.

Coalition formation involves three activities (Tohme and Sandholm [1999]): coalition structure generation (partitioning the agents into disjoint coalitions), solving each coalition's (optimization) problem within the coalition, and dividing the value of each coalition among member agents (in case of net cost, this value may be negative). Coalition formation among self-interested agents has been widely studied in game theory. The main solution concepts are geared toward payoff division among agents in ways that guarantee forms of stability of the coalition structure.

Contract-nets

An agent may try to contract out some of the tasks that it cannot perform by itself, or that may be performed more efficiently by other agents. One self-interested agent may convince another self-interested agent to help it with its task, by promises of rewards. In the contract-net protocol[2] (Smith [1988]), an contract is an explicit agreement between an agent that generates a task and an agent that is willing to execute the task. In this kind of protocol there are two types of agent: managers (initiators) and the contractors (participants). New tasks to be executed by the system are given to one of the managers. Typically, the manager broadcasts a request for proposals to all the available contractors. The contractors then respond to the manager by making a proposal based upon their capabilities or they refuse should they decide that they lack the resources needed to perform the task. After some time has elapsed, the manager chooses from the proposals received, the best offer and subsequently gives the task to be executed to the corresponding contractor. After that the chosen contractor reports back to manager with the results of the executed task or a notification of its failure.

3.1.3 Negotiation Techniques

Apart the specific protocols of negotiation, the design of applications based on negotiation involves also the specification of strategies and agents' decision making models. These involves a sort of rules from different approaches, such as game-theory, heuristics, and argumentation. In this section such techniques are commented.

[2]FIPA contract net protocol specification: http://www.fipa.org/specs/fipa00029/SC00029H.html

3.1.3.1 Game-theory models

Game theory is a branch of economics that studies interaction between self-interested agents. The techniques from game theory are applied to interaction that occur between agents which are regarded as self-interested utility maximizers (Rosenschein and Zlotkin [1994], Kraus [1997]). The models of game theory are abstract representation of classes of situations that involve individuals who have different goals and preferences. Such abstract models can be used as a basis for the agents' interactions protocols, where agents are modeled as players of game-theoretic models.

According to (Jennings et al. [2001]), game theoretic techniques can be applied to two key problems: design of an appropriate *protocol* that will govern the interactions between the negotiation participants; and design of a particular *strategy* (the agents' decision making models) that individual agents can use while negotiating - an agent will aim to use a strategy that maximizes its own individual welfare. In order for an agent to make the choice that optimizes its outcome, it must reason *strategically*, i.e., it must to take into account the decisions that other agents may make, and must assume that they will act so as to optimize their own outcome. In negotiation, this means, for example, taking into account the private valuations that agents have of the negotiation issues, their private deadlines for making a deal, and so on. Game theory gives a way of formalizing and analyzing such concerns.

There is one important limitation of the game-theoretic approach, i.e., searching the solution in exhaustive fashion. Considering the limitation of computational power, many heuristic techniques are adopted to develop new models, namely heuristic-based negotiation models (Winoto et al. [2002]). Using these models, the negotiators can make decisions faster to find a good solution, although not necessarily the best one.

3.1.3.2 Heuristic approaches

The heuristic approaches (Faratin et al. [1998], Barbuceanu and Lo [2000]) can be seen as seen either as computational approximations of game theoretic techniques or they may be computational realizations of more informal negotiation models.

The central concern of this approach is to heuristically model the agent's decision making during the course of the negotiation (Lomuscio et al. [2001]). The chosen negotiation protocol is a repeated, sequential model where offers are iteratively exchanged. The space of possible agreements is quantitatively represented by contracts having different values for each issue. Each agent then rates these points in the space of possible outcomes according to some preference structure, captured by an utility function. Proposal and counter-proposals are then offers over single points in this space of possible outcomes, and the search terminates either when the time to reach an agreement has been exceeded or when a mutually acceptable solution, an intersection in the joint space of possible outcome of both agents, has been

reached.

The offers and counter offers can be also generated by linear combinations of simple functions, called *tactics* (Faratin et al. [1998]). Tactics generate an offer, or counter-offer, for a single component of the negotiation object using a single criteria (time, resources, etc). Different weights in the lineal combination allow the varying importance of the criteria to be modeled. A *strategy* denotes the way in which an agent changes the weights of the different tactics over time. *Tactics* are the set of functions that determine how to compute the value of an issue (price, quality) by considering a single criteria. The set of values for the negotiation issue are then the range of the function, and the single criteria is its domain. Given that agents may want to consider more than one criterion to compute the value for a single issue, the generation of counter proposal is modeled as a weighted combination of different tactics covering the set of criteria. The values so computed for the different issues will be elements of the counter proposal. For instance, if an agent wants to counter propose taking into account two criteria: the remaining time and the previous behavior of the opponent, it can select two tactics: one from the time-dependent family and one from the imitative family. Both of these tactics will suggest a value to counter propose for the issue under negotiation. The actual value which is counter proposed will be the weighed combination of the two suggested values.

Heuristic models need extensive evaluation, typically through simulations and empirical analysis, since it is usually impossible to predict precisely how the system and the constituent agents will behave in a wide variety of circumstances.

3.1.3.3 Argumentation-based approaches

Argumentation may be used both at the level of an agent's internal reasoning and at the level of negotiation between the agents. At the negotiation level, the basic idea behind the argumentation-based approach is to allow additional information to be exchanged, over and above proposals (Jennings et al. [2001]). This information can be of a number of different forms, all of which are arguments which explain *explicitly* the opinion of the agent making the argument. Thus, in addition to rejecting a proposal, an agent can offer a critique of the proposal, explaining why it is unacceptable. Similarly, an agent can accompany a proposal with an argument which says why the other agent should accept it.

Arguments are meta-information that summarize the reasons why a proposal should be accepted. Common categories of arguments include: threats (failure to accept this proposal means something negative will happen to you), rewards (acceptance of this proposal means something positive will happen to you), and appeals (you should prefer this option over that alternative for some reason). Moreover, arguments can be seen as a series of logical steps (Parsons and Jennings [1996a]) for and against propositions of interest, and the weight given to an argument may be determined by examining the support for the steps in the argument.

According to (Parsons et al. [1998], Jennings et al. [1998]), the role of the argument may be twofold:

- It allows agents to justify their negotiation stance: an agent might have a compelling reason for adopting a particular negotiation stance (for example, a company may not be legally entitled to sell a particular type of product to a particular type of consumer). In such cases, the ability to provide the justification for its attitude towards a particular issue can allow the opponent to more fully appreciate an agent's constraints and behavior.
- It can be the basis which agents persuade one another to change their negotiation stance: agents sometimes need to actively change their opponents' agreement space, or its rating over that space, in order for a deal to be possible. In such cases, agents seek to construct arguments that they believe will make their opponent look more favorable upon their proposal.

An argument can be seen also as sequence of inferences leading to a conclusion. It the argument is correct, the conclusion is true. For example, an agent x does not agree with the proposal a of the agent y if, when x can build an argument $\neg a$ that rebuts the initial proposal. The explanation for why a particular proposal is made is the argument that supports it. Once the proposal is made it is evaluated by other agents through argumentation by the device of seeing whether the argument for the proposal may be defeated either because it rebuts an objective or because it undercuts or is undercut by the argument for achieving the other agent's objective If this kind of objection is detected the argument created by the other agent will serve as a critique of the initial proposal A counter proposal may then be generated by either the original agent or the responding agent using the information that form the proposing and or critiquing arguments as a guide to what is likely to be acceptable Thus the use of argumentation also providing meta information. The notion of acceptability of arguments provides a way of rating possible suggestions to ensure that only the best is sent.

3.2 Argumentation

As commented in the previous sections, argumentation can be seen as a more sophisticated kind of negotiation or a model for reasoning based on the construction and comparison of arguments. In this section, it is focused on the second sense of the interpretation of the term argumentation. Here, argumentation is seen as the interaction of different arguments for and against some conclusion. More specifically, argumentation is a reasoning model based on the construction of arguments and counter-arguments followed by the selection of the most acceptable of them (Amgoud and Cayrol [2002b]).

In Multi-Agent Systems, argumentation has been applied as a way to facilitate rational interaction. A single agent may also use argumentation techniques to perform its individual reasoning because it needs to make decisions under complex preferences policies, in a highly dynamic environment. Argumentation provides tools for designing, implementing and analyzing sophisticated forms of interaction among rational agents. Argumentation has made solid contributions to the practice of multi-agent dialogues. As commented before, there is s fine line separating negotiation, dialogues, and argumentation.

Applications for this view on argumentation include: legal disputes, business negotiation, labor disputes, team formation, scientific inquiry, deliberative democracy, ontology reconciliation, risk analysis, scheduling, and logistics.

In this section, the main argumentations frameworks proposed in the literature to deal with the selection of acceptable arguments are presented.

3.2.1 Argumentation Frameworks

An argumentation system is generally composed of five elements (Prakken and Vreeswijk [2000]): (1) a logical language; (2) an argument definition; (3) a concept of conflict among arguments; (4) a concept of defeated argument; and (5) a concept of argument acceptability. The definition of an argument depends of the application: an argument can be seen as a plan for an agent to achieve or a sequence of chained implicative rules.

The central notion in argumentation systems is the notion of *acceptability*, which has been most often defined on basis of defeaters. The resulting evaluation of arguments is based on the interactions between defeaters. The different approaches developed for reasoning in argumentation systems use either the individual acceptability (Elvang-Gand Hunter [1995]) or the joint acceptability (Dung [1995]):

- *individual acceptability*: an acceptability level is assigned to a given argument on the basis of the existence of direct defeaters.
- *joint acceptability*: the set of all arguments that a rational agents accepts must defend itself against any defeater.

A most classical argumentation framework was proposed by (Dung [1995]). Several systems, such as the Value-Based Argumentation Framework (Bench-Capon [2003]) and the Preference-Based Argumentation Framework (Amgoud and Cayrol [2002a], Amgoud and Cayrol [2002b]) use as basis the classical framework of Dung. Moreover, (Prakken and Sartor [1997]) have extended Dung's framework with priorities. They present a language with defeasible and strict rules, and strong negation (a sort of classical negation) and weak

negation (a negation by failure), before considering the use of priorities and defeat. The system is inspired by some features of the legal reasoning.

In this section, the most promising argumentation frameworks proposed in the literature are detailed.

3.2.1.1 Classical argumentation framework

According to Dung (Dung [1995]), a basic argumentation framework is defined as a pair consisting of a set of arguments and a binary relation representing the defeasibility relationship between arguments. An argument is an abstract entity whose role is determined by its relation to other arguments. Dung defines an argumentation framework as follows.

Definition 1 (Dung [1995]) An Argumentation Framework is a pair $AF = (AR, attacks)$, where AR is a set of arguments and *attacks* is a binary relation on AR, i.e., $attacks \subseteq \text{AR} \times \text{AR}$. An *attack*(A,B) means that the argument A attacks the argument B. A set of arguments S attacks an argument B if B is attacked by an argument in S.

The key question about the framework is whether a given argument A, $A \in AR$, should be accepted. One reasonable view is that an argument should be accepted only if every attack on it is rebutted by an accepted argument (Dung [1995]). This notion produces the following definitions:

Definition 2 (Dung [1995]) An argument $A \in AR$ is *acceptable* with respect to set arguments S, *acceptable(A,S)*, if $(\forall\ x)(x \in AR) \wedge (attacks(x,A)) \longrightarrow (\exists\ y)(y \in S) \wedge attacks(y,x)$

An argument is acceptable with respect to a set S of arguments if it is defended by that S against all its defeaters.

Definition 3 (Dung [1995]) A set S of arguments is *conflict-free* if $\neg(\exists\ x)(\exists\ y)((x \in S) \wedge (y \in S) \wedge attacks(x,y))$

Definition 4 (Dung [1995]) A conflict-free set of arguments S is *admissible* if $(\forall x)(x \in S) \longrightarrow acceptable(x,S)$

Definition 5 (Dung [1995]) A set of arguments S is a *preferred extension* if it is a maximal (with respect to inclusion set) admissible set of AR.

Figure 3.1: Arguments and attacks in an AF.

A *preferred extension* represent a consistent position within *AF*, which can defend itself against all attacks and which cannot be further extended without introducing a conflict.

According to (Amgoud and Cayrol [2002a]), in a given argumentation framework, three categories of arguments are distinguished:

- the *non-defeated* arguments, which are gathered in the *class of acceptable arguments*, and represent the "good" arguments.
- the *arguments defeated* (in the sense of the relation *attack*), which are gathered in the *class of rejected arguments*.
- the arguments that are neither acceptable nor rejected, and which are gathered in the *class of arguments in abeyance*.

Formally, these classes are defined as follows.

Definition 6 (Amgoud and Cayrol [2002a]) Let (AR,attacks) be an argumentation framework. The *class of acceptable arguments*, $\text{Acc}_{attacks}$, is the set $\{x \in AR \mid$ there does not exist $y \in AR$ such attacks(A,B) $\}$. The class of rejected arguments, $\text{Rej}_{attacks}$, is denoted by $\{x \in AR \mid \exists\, y \in \text{Acc}_{attacks}$ such that attacks(B,A) $\}$. The class of arguments in abeyance is $\text{Ab}_{attacks} = AR \setminus (\text{Acc}_{attacks} \cup \text{Rej}_{attacks})$

Consider an argumentation framework AF = (AR,attacks), where AR = A,B,C and attacks = (A,B), (B,C). A useful way to see an AF is as a directed graph, as shown in Figure 3.1, in which the arguments are vertices and the attacks are represented as edges, directed from attacker to attacked. The argument A is not defeated, and the arguments C is defended by the argument A, i.e., $\text{Acc}_{attacks}$ = A,C. The argument B is defeated by A, which is acceptable, so $\text{Rej}_{attacks}$ = B. $\text{Ab}_{attacks} = \emptyset$.

3.2.1.2 Value-based argumentation framework

In Dung's frameworks, attacks always succeed. (Bench-Capon [2003]) propose an extension to model of Dung, Value-Argumentation Framework (VAF), in order to allow associate arguments with the social values they advance. Then, the attack of one argument on another

Figure 3.2: Arguments and attacks in a VAF.

is evaluated to say whether or not it succeeds by comparing the preferences of the values advanced by the arguments concerned.

In many domains, arguments provide reasons which may be more or less persuasive (Laera et al. [2006]). Their persuasiveness may vary according to their audience. The VAF is able to distinguish attacks from successful attacks, those which defeat the attacked argument, with respect to an ordering on the preferences that are associated with the arguments. It allows accommodate different audiences with different interests and preferences.

Definition 1 (Bench-Capon [2003]) A Value-based Argumentation Framework (VAF) is a 5-tuple $VAF = (AR, attacks, V, val, P)$ where $(AR, attacks)$ is an argumentation framework, V is a nonempty set of values, val is a function which maps from elements of AR to elements of V and P is a set of possible audiences. For each $A \in AR$, $val(A) \in V$.

For example, consider a VAF = (AR,attacks,V,val,P), where AR = A,B,C, attacks = (A,B), (B,C), V = syntactic,semantic, where for the audience *syntactic* the value *syntactic* is preferred to the value *semantic*, and for the audience *semantic*, the value *semantic* is preferred to the value *syntactic*. Such arguments could be derived from syntactic and semantic matching approaches. The VAF is shown in Figure 3.2.

Definition 2 (Bench-Capon [2003]) An Audience-specific Value Based Argumentation Framework (AVAF) is a 5-tuple $VAF_a = (AR, attacks, V, val, valpref_a)$ where $AR, attacks, V$ *and* val are as for a VAF, a is an audience and $valpref_a$ is a preference relation (transitive, irreflexive and asymmetric) $valpref_a \subseteq V \times V$, reflecting the value preferences of audience a. $valpref(v_1, v_2)$ means v_1 is preferred to v_2.

Definition 3 (Bench-Capon [2003]) An argument $A \in AR$ $defeats_a$ (or *successfully attacks*) an argument $B \in AR$ for audience a if and only if both $attacks(A,B)$ and not $valpref(val(B), val(A))$.

An attack succeeds if both arguments relate to the same value, or if no preference between the values has been defined. If V contains a single value, or no preference between the values has been defined, the AVAF becomes a standard AF. If each argument can map to a different value, a Preference Based Argumentation Framework is obtained (Amgoud and Cayrol [1998]), as described in the next section.

Definition 4 (Bench-Capon [2003]) An argument $A \in AR$ is *acceptable* to audience a ($acceptable_a$) with respect to set of arguments S, $acceptable_a(A,S)$ if $(\forall x)((x \in AR \wedge defeats_a(x,A)) \longrightarrow (\exists y)((y \in S) \wedge defeats_a(y,x)))$.

In the example above, the argument A is acceptable in both audiences because it is not attacked. For the audience *syntactic*, the argument B is acceptable considering that B is preferred to A (which belongs to the audience *semantic*). On the other hand, for the audience *semantic*, the argument C is acceptable, due the fact that C is defended by A, which has greater preference.

Definition 5 (Bench-Capon [2003]) A set S of arguments is *conflict-free* for audience a if $(\forall x)(\forall y)((x \in S \wedge y \in S) \longrightarrow (\neg attacks(x,y) \vee valpref(val(y),val(x)) \in valpref_a))$.

Definition 6 (Bench-Capon [2003]) A *conflict-free* set of argument S for audience a is *admissible* for an audience a if $(\forall x)(x \in S \longrightarrow acceptable_a(x,S))$.

Definition 7 (Bench-Capon [2003]) A set of argument S in the VAF is a *preferred extension* for audience a ($preferred_a$) if it is a maximal (with respect to set inclusion) *admissible* for audience a of AR.

The significance of theses notions is with reference to which arguments in the VAF it is possible consistently to accept (Bench-Capon [2002]). If an argument is acceptable with respect ro AR, it cannot be defeated and thus must be acceptable. Similarly, is an argument is acceptable with respect to a subset S of AR, it is possible to accept it, provided by S is accepted. S cannot be consistently accepted unless it is conflict free. Thus, an admissible set is one that can be consistently accepted given a VAF. If, however, an admissible set can be further extended, it is possible to accept further arguments. Thus, a preferred extension represents a set of acceptable arguments to which no more arguments can be added.

Considering the given example, the preferred extension for the audience *syntactic* is $\text{pref}_{syntactic}$ = {A,B}, while the preferred extension for the audience *semantic* is $\text{pref}_{semantic}$ = {A,C}.

In order to determine the preferred extension with respect to a value ordering promoted by distinct audiences, (Bench-Capon [2003]) introduces the notion of *objective* and *subjective* acceptance.

Definition 8 (Bench-Capon [2003]) An argument $x \in AR$ is *subjectively* acceptable if and only if x appears in the preferred extension for some specific audiences but not all. An argument $x \in AR$ is *objectively* acceptable if and only if, x appears in the preferred extension for every specific audience. An argument which is neither objectively nor subjectively acceptable is said to be *indefensible*.

Following the example, the argument A is *objectively acceptable*, while the arguments B and C are *subjectively acceptable*.

The set of *objectively acceptable* arguments represents the consensus between the parties. In (Bench-Capon [2002]), such set is called *sceptically acceptable*. If an argument is not in this consensus, it can considered at least defensible – subjectively accepted so called *credulously acceptable* in (Bench-Capon [2002]).

3.2.1.3 Preference-Based Argumentation Framework

Dung's definition of acceptability disregards the quality of the arguments. The force of an argument can be often estimated by considering the beliefs used to build this argument. According with the preferences that can exist between the beliefs, an argument can be more or less strong than another argument.

(Amgoud and Cayrol [1998], Amgoud and Cayrol [2002b], Amgoud and Cayrol [2002a]) refines the classical argumentation framework proposed by Dung, taking into account preference relations between arguments in order to integrate two complementary points of view on the concept of acceptability: acceptability based on the existence of direct counter-arguments and acceptability based on the existence of defenders. An argument is thus acceptable if it is preferred to its direct defeaters or if it is defended against its defeaters. So, two complementary notions of defense are used: *individual defence* (introduced by preference relations), and the notion of defence proposed by Dung, which may be called *joint defence*. The result of combining preference relations and defeasibility relations leads to a Preference-based Argumentation Framework (PAF):

Definition 1 (Amgoud and Cayrol [2002a]) A *Preference-based Argumentation Framework* (PAF) is a triple (AR,attacks,Pref), where A is a set of arguments, attacks is a binary relation representing a defeat relation between arguments, attacks $\subseteq$ A $\times$ A, and Pref is a (partial or total) preordering on A $\times$ A.

Instead of keeping only the arguments that are not defeated, the arguments that are preferred to their defeaters are also accepted. It is said that such an argument *defends itself* (*individual defense*) against all attacks.

Definition 2 (Amgoud and Cayrol [2002a]) Let (AR,attacks,Pref) be a PAF. Let A, B be two arguments of A such that attacks(B,A). A *defends itself* against B (wit respect to Pref) iff A $\gg_{Pref}$ B. An argument defends itself (with respect to Pref) iff it is preferred with respect to Pref to each of its defeaters.

$C_{attacks,Pref}$ denotes the set of arguments defending themselves (with respect to Pref) against their defeaters.This set contains also the arguments which are not defeated. Each argument in $C_{attacks,Pref}$ is an acceptable argument. This corresponds to the individual point of view.

However, $C_{attacks,Pref}$, is restricted since it discards arguments which appear acceptable. Considering an argument defeated by B such that B is preferred to A. It is clear that A does not belongs to $C_{attacks,Pref}$. Assuming that B itself is defeated by an argument C which is preferred to B. A might be regarded as an acceptable argument. This corresponds to the *joint defense* point of view:

Definition 3 (Amgoud and Cayrol [2002a]) An argument is *defended* by S (with respect to Pref) iff $\forall$ B $\in$ A, if attacks(B,A) and not (A $\gg_{Pref}$ B) then $\exists$C $\in$ S such that attacks(C,B) and not (B $\gg_{Pref}$ C).

The acceptable arguments are the ones which defend themselves against their defeaters ($C_{attacks,Pref}$) and also the arguments which are defended (directly or indirectly) by the arguments of $C_{attacks,Pref}$.

In a PAF, the acceptability concept is then defined as follows:

Definition 4 (Amgoud and Cayrol [2002a]) The *class of acceptable arguments*, $Acc_{attacks,Pref}$, is the set {A $\in$ A | $\forall$B $\in$ A if attacks(B,A) then A $\gg^{Pref}$ B }. The *class of rejected arguments*, $Rej_{attacks,Pref}$, is {A $\in$ A | $\exists$ B $\in$ $Acc_{attacks,Pref}$ such that attacks(B,A) and not (A $\gg^{Pref}$B) }. The class of arguments in abeyance is $Ab_{attacks,Pref}$ is AR $\setminus$ ($Acc_{attacks,Pref} \cup Rej_{attacks,Pref}$).

Considering (AR,attacks,Pref) (Figure 3.3) be a PAF defined by AR = A,B,C,D, attacks = (A,B),(C,D),(D,C), and C $\gg^{Pref}$ D. In the classical argumentation framework of Dung, the argument A is not defeated, i.e., $Acc_{attacks}$ = A; the argument B is defeated by A, which is acceptable, i.e., $Rej_{attacks}$ = B; the two arguments C and D defeat each other, so neither C nor D is acceptable, i.e., $Ab_{attacks}$ = C,D. In PAF, the class of acceptable arguments is $Acc_{attacks,Pref}$ = A,C, where the argument A is not defeated and C is defended by itself; $Rej_{attacks,Pref}$ = B,D, and $Ab_{attacks,Pref}$ = $\emptyset$.

3.3 Summary

In this chapter, approaches based on negotiation and argumentation, in the context of Multi-Agent systems were presented. There is a fine line of separation between negotiation and argumentation. Negotiation involves different forms of communication between agents that

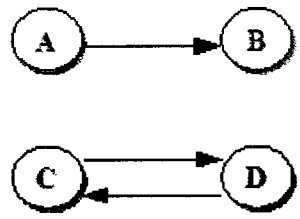

Figure 3.3: Arguments and attacks in a PAF.

iteratively exchange proposals and counterproposals in order to solve its tasks. Argumentation can be seen as a more sophisticated exchange of deals in a negotiation protocol. Moreover, argumentation provides also a model for reasoning based on the construction and comparison of arguments. The different protocols of negotiation and the main argumentation frameworks proposed in the literature were commented.

Regarding the argumentation frameworks, the Dung's proposal does not consider the strength of arguments in the the notion of acceptability of arguments. (Bench-Capon [2003]) and (Amgoud and Cayrol [2002a]) extend the Dung's proposal in order to define different preference relations between arguments. (Bench-Capon [2003]) proposes local definition of preference, where arguments are preferred to each other according to a specific audience, while (Amgoud and Cayrol [2002a]) specifies a notion of global preference, where arguments can defend itself against all other weaker arguments.

In the field of ontology matching, negotiation and argumentation can be seen as promising approaches to combine matching techniques. Agents representing different matching approaches can try to arrive to an alignment consensus on their different views of the problem using such approaches. In the next section the main related work on ontology matching are presented, including the recent proposals in using negotiation and argumentation.

Chapter 4

Ontology Matching Systems

The diversity of ontology matching techniques and strategies have been demonstrated in a great variety of matching systems. In this chapter, a comparative study of the main ontology matching systems proposed in the literature is presented. These systems are classified according the kind of approaches they use, in two groups: *standard* matching systems and *cooperative agent-based* systems.

The chapter is organized as follow. Section 4.1 presents the *standard* ontology matching systems. Section 4.2 comments on the negotiation and argumentation-based matching systems, which are closely related with the proposal presented in this thesis. Finally, Section 4.3 presents the summary of the chapter in the context of the thesis.

4.1 Standard Ontology Matching Systems

The ontology matching systems presented in this section are classified according to the kind of information, *ontology-level* or *data-level*, used in the matching process. This classification is inspired on the proposal of (Euzenat and Shvaiko [2007]). First, the systems that focus on the use of ontology-level information are discussed. Next, the systems that exploit instances of the ontologies are presented. Third, the systems that exploit both ontology-level and data-level information are commented.

4.1.1 Ontology-level information

Table 4.1 presents a review of the systems that are based on the use of ontology-level information, such as labels of classes and properties, and ontology hierarchy. Such systems are compared using four parameters: kind of input, technique applied in the matching process,

strategy used to combine the techniques, and target operation. The techniques and strategies are characterized according to the classifications provided in the Sections 2.2.4.2 (Basic Techniques) and 2.2.4.3 (Matching Strategies), respectively. The detailed description of each system can be found in the Appendix B (the respective section of the system is indicated below of its name).

As shown in Table 4.1, when considering the technique used in the matching process, the majority of the systems apply string-based techniques, which range from simple comparison of strings (they are totaly similar or not), such as applied by (Hovy [1998]) to edit-distance similarity (such as in COMA, S-Match, ASCO, DSSim, PriorPlus, and XSOM), TFIDF measure (such as in ASCO), and Jaccard measure (DSSim). The other systems apply prefix and suffix comparison (Similarity Flooding) and n-gram method (S-Match). Moreover, several systems combine the string-based approaches with the synonymous relations provided by a thesauri, commonly the WordNet (Skat, Dike, Artemis, Coma, XClust, S-Match, MoA, ASCO, HCone, DSSIm, and X-SOM). Regarding the structural approaches, several heuristics are used, such as number of common descendants (Hovy [1998], Cupid, XClust, PriorPlus), number of similar nodes in the path between the root and the element (Anchor-Prompt, XClust, S-Match, PriorPlus), neighbors (ASCO and H-Match).

A variety of strategies to combine individual matching techniques is used in the systems. The techniques can be executed in parallel (H-Match, Dssim, XSOM, and PriorPlus for instance), sequentially (Milo and Zohar [1998], SKAT, DIKE, Artemis, Anchor-Prompt, OntoBuilder, MOA, and TaxoMap), or in a hybrid manner (parallel at element-level and sequential at structural-level, such as done by S-Match). To combine the results of these executions, several ways are proposed: simple weighted formula (Hovy [1998], Cupid, and PriorPlus), average of the results (COMA), threshold to select the best matcher (COMA), linear combinations (H-Match), hierarchical clustering (XClust), Dempster's rule of combination (DSSim), and combination using a feed-forward neural network (XSOM).

Few systems explicitly define a pre-processing phase. For instance, OntBuilder and ASCO include a pre-processing to remove stop terms and noisy characters. MoA, ASCO and H-Match apply the tokenization of strings, in order to deal with composite strings. TaxMap defines semantic relations based on the matching of tokens of the labels (for instance, to labels are equivalents if they have exactly the same string, or they are related by a subsumption relation if there is inclusions of labels). (Hovy [1998]) propose a score associated with composite words: *namescore* = square of number of letter matched + 20 points if words are exactly equal or 10 points if end of match coincides.

Regarding the reuse of mappings, just COMA proposes a reuse-oriented matcher (COMA), which uses previously obtained results for new schemas.

The interaction with the user is proposed in some systems: mappings rules provided by domain experts (SKAT), pairs of related terms can be defined by the user (Anchor-Prompt),

or different levels of user interaction (fully automated, semi-automated, and user-based, such as proposed by HCONE).

The output of the majority of systems is a score of similarity in the range [0,1]. Only S-Match, HCONE and TaxoMap explicitly specify as output semantic relations such as equivalence and subsumption, while DSSim and OntoDNA output boolean values.

[ht]

Table 4.1: Comparison of the ontology-level based systems.

System	Input	Technique	Strategy	Operation
(Hovy, 1998) A.1.1	taxonomy	string-based, taxonomy-based	parallel similarity aggregation	integration
(Milo and Zohar, 1998) A.1.2	SGML, OO	thesauri, graph-based	sequential	translation
SKAT (1999) A.1.3	RDF	string-based, graph-based	sequential	translation
ONION (2000) A.1.3	RDF	string-based, thesauri, graph-based	sequential	translation
DIKE (2000) A.1.4	ER	thesauri, graph-based	sequential	query mediation
Artemis (2001) A.1.5	relational schema OO, ER	string-based, thesauri	sequential	query mediation
Anchor-Prompt (2001) A.1.6	OWL, RDF	string-based, graph-based	sequential	merging
OntoBuilder (2001) A.1.7	XML schema	string-based thesauri	sequential	query mediation
Cupid (2001) A.1.8	XML, relational schemas	string-based, thesauri, graph-based	sequential similarity aggregation	
COMA (2002) A.1.9	XML, relational schemas OWL	string-based, thesauri, reuse, taxonomy-based	parallel similarity aggregation	translation
Similarity flooding (2002) A.1.10	XML, relational schemas	string-based, graph-based	sequential	
XClust (2002) A.1.11	DTD	thesauri, constraints, graph-based	sequential similarity aggregation	integration
S-Match (2004) A.1.12	XML schema, OWL	string-based, thesauri, graph-based, formal (SAT)	parallel (element-level), sequential	
MoA (2005)	OWL	pre-processing, taxonomy-based,	sequential	

Table 4.1: Comparison of the ontology-level based systems.

System	Input	Technique	Strategy	Operation
A.1.13		thesauri		
ASCO (2005) A.1.14	RDFS, OWL	pre-processing, string-based, thesauri, graph-based	sequential similarity aggregation	
OMEN (2005) A.1.15	OWL	string-based, graph-based	sequential	
H-Match (2006) A.1.16	OWL	thesauri, taxonomy-based	parallel, linear combination	P2P, query mediation
MapOnto (2006) A.1.17	XML schema, relational schema	graph-based, manual	sequential	translation
HCONE (2006) A.1.18	OWL	thesauri, user-input		merging
DSSim (2007) A.1.19	OWL	string-based, thesauri, graph-based	parallel similarity aggregation	query mediation
OntoDNA (2007) A.1.20	OWL	pre-processing, string-based, statistical, machine learning	sequential	
PriorPlus (2007) A.1.21	XML, OWL	string-based taxonomy-based	parallel (similarity aggregation), Constraint Satisfaction Solver	
TaxoMap (2007) A.1.22	taxonomy	pre-processing, string-based		
X-SOM (2007) A.1.23	RDF, OWL	string-base, graph-based, thesauri, contextual knowledge	parallel similarity aggregation	

4.1.2 Data-level information

The data-level systems are based uniquely on the instances comparison. Table 4.2 shows a review of such systems. Comparing values of attributes is commonly adopted in database integration, and as shown in Table 4.2, part of these systems receive as input populated relational schemas (LSD, AutoMatch, GLUE, iMap, Dumas, and sPLMap) to discover correspondences between attributes of two schemas (such as done in iMap and AutoMatch). Moreover, matching in these systems aim to exchange documents annotated with the ontologies (CAIMAN), merge ontologies based on instances (FCA-Merge), or determine correspondences between classes of two classification by comparing the membership of the documents od these classes (SBI).

Basically, statistical and machine learning techniques are applied: TFIDF (CAIMAN and Dumas), Formal Concept Analysis (FCA-Merge), bayesian learning (AutoMatch, iMAP), minimal information and entropy (Kang and Naughton [2003], multi-strategy learning approach (GLUE), K-statistic (SBI), and Jaccard similarity to measure the overlap between set of instances (Isaac et al. [2008b]). Moreover, systems such as iMAP, sPLMap and Dumas use based string-based techniques to compare values of attributes.

4.1.3 Ontology and data level information

Table 4.3 presents the review of the main systems that use both ontology-level and data-level information. Several systems execute different matchers (i.e., string-based, structure-based, and instance-based) in parallel and combine their results: rules heuristically defined (NON), machine learning classifiers (C4.5 as done in Embley et al. [2004]), systems of equations (OLA), linear interpolation (RiMON), formal concept analysis to extract alignments based both on maps of concepts and instances (IF-MAP), weighted average (oMap), weighted sum

Table 4.2: Comparison of the data-level based systems.

System	Input	Technique	Strategy	Operation
CAIMAN (2001) A.2.1	classification	ML		document exchange
FCA-Merge (2001) A.2.2	ontology	Formal Concept Analysis (FCA)		merging
LSD (2001) A.2.3	XML schema, relational schema	ML		integration
Automatch (2002) A.2.4	relational schema	ML		
GLUE (2004) A.2.6	XML, relational schemas	ML		
iMAP (2004) A.2.7	relation schema	string-based, statistical	sequential	
SBI (2004) A.2.8	classification	statistical		
(Kang and Naughton, 2003) A.2.5	relational schema	statistical		
DUMAS (2005) A.2.9	relational schema	string-based, statistical		
sPLMap (2005) A.2.10	database schema	string-based ML	similarity aggregation	translation
(Isaac et. al., 2007) A.2.11	thesauri	statistical		

(SAMBO), and experimental weighted (Lily). Some systems combine sequentially instance-based attribute classification with edit-distance string-based (such as done in Clio). Falcon execute sequentially a TFIDF linguistic matcher that combine concepts and instances, and a graph-based matchers. Moreover, ASMOV iteratively combines several matchers using a single weighted sum to combine the individual results. A combination of names and values of attributes is used as input to a neural network to learn rules of mapping (SEMINT). Instance-based matcher are commonly based on Naive-Bayes classifiers (Clio, RiMON, oMap), statistics (SEMINT, NON, Falcon, Silas, SEMA), probabilistic methods(SAMBO), pattern extraction (Madhavan et al. [2005]).

Regarding the kind of output, Lily, Sema, Silas, and Sambo produce boolean values for the equivalence relation, while oMap is able to return equivalence and subsumption relations. The other systems output the standard confidence values.

Table 4.3: Comparison of the ontology and data-level based systems.

System	Input	Technique	Strategy	Operation
SEMINT (2000) A.3.1	relational schema	constraint, data level (statistical, ML)	sequential	integration
Clio (2000) A.3.2	XML, relational schema	string-based, data level (ML)	sequential	translation
IF-Map (2003) A.3.3	KIF, RDF	string-based, taxonomy-based data-level (FCA)		
NON (2004) A.3.4	OWL, RDF	string-based, taxonomy-based data-level (statistical)	parallel, similarity aggregation	
QOM (2004) A.3.5	OWL, RDF	string-based, taxonomy-based data-level (statistical)	parallel, similarity aggregation	
(Embley et. al., 2004) A.3.6	XML schema, taxonomy	string-based, thesauri, data-level (statistical)	parallel, similarity aggregation	
OLA (2004) A.3.7	OWL, RDF	string-based, graph-based, data-level (statistical)	parallel, similarity aggregation	
RiMON (2004) A.3.8	OWL	string-based, thesauri, taxonomy-based, data-level (statistical)	parallel (similarity aggregation), sequential (iteration)	
oMap (2005) A.3.9	OWL	string-based, formal data-level (ML)	parallel (similarity aggregation), sequential	query answering
(Madhavan et. al., 2005) A.3.10	relational schema	string-based data-level (statistic)	parallel	- -
FALCON-AO (2006) A.3.11	OWL, RDF	string-based, graph-based data-level (statistics)	sequential similarity aggregation	-
Silas (2007) A.3.12	thesauri	string-based data-level (statistic)	sequential	-
ASMOV (2007) A.3.13	OWL	string-based, thesauri graph-based, constraints,	parallel, similarity aggregation (weighted sum), iteration	integration
Lily (2007) A.3.14	RDF	string-based, graph-based	parallel, similarity aggregation (weighted sum)	

Table 4.3: Comparison of the ontology and data-level based systems.

System	Input	Technique	Strategy	Operation
SAMBO (2007) A.3.15	OWL	string-based, thesauri, graph-based, data-level (ML)	parallel, similarity aggregation (weighted sum)	
SEMA (2007) A.3.16	OWL	string-based, graph-based, thesauri, data-level (string-based)	sequential (iterative)	

4.1.4 Comparison of Standard Matching Systems

As presented in the previous sections, the matching systems represent a great variety of techniques and strategies. Similar methods are used by these systems, but the strategies used to combine such methods vary (for example, by simply changing the order of execution of the methods can generated different results).

A comparative evaluation of these systems is hard to do, due the fact that they use so different experimental tests. Recently, the Ontology Alignment Evaluation Initiative (OAEI)[1] has done efforts to establish a consensus for evaluation of the methods for ontology matching. It organizes evaluation campaigns that aims to evaluate ontology matching technologies, through the controlled experimental evaluation of the techniques performances. In this section, the comparative results reported in the OAEI campaigns are presented.

The first complete set of tests was presented in the OAEI 2005 (Euzenat et al. [2005]), where two set of tests were used: *benchmark* (containing 53 tests using ontologies of bibliography domain) and *directory* (formed by web directories from Google and Yahoo). For *benchmark* track, three described systems are considered, namely Ola, Falcon, and oMap. Falcon seems largely dominant. For *directory* track, which is a very hard task, Falcon and OLA produced similar values of recall (about 30%), representing better values than oMap.

The OAEI 2006 campaign was formed by a greater number of data sets and participants (see Euzenat et al. [2006] for details). For this campaign, Falcon, H-Match, DSSim, COMA, Prior, and RiMON were evaluated (not all systems participate of all tests). Five set of tests were considered, namely[2]: *benchmark*, like the previous campaign, *anatomy*, covering the real world ontologies of body anatomy domain, *directory*, like the previous campaign, *food*, containing two thesauri about food, and *conference*, composed by a collection of conference

[1] http://oaei.ontologymatching.org/

[2] *Jobs* track was not performed due the technical problems.

organization ontologies. Only *food* tests consider three semantic relations (exactMatch, narrowMatch, and broadMatch).

For *benchmark* tests, three systems were relatively close (COMA, Falcon, and RiMON). The RiMON system was slightly ahead of the others systems. No systems had strictly lower performance than edna, a very simple edit distance algorithm.

For *anatomy* tests, due to the lack of a reference mapping, the evaluation was concentrated on the coverage of the ontologies, treatment of irregular concepts, and applied techniques. None of the systems had a good coverage of the ontologies and only COMA was able to determine mappings for irregular concept names – however, it was not able to match any of the regular concept names. Looking at the methods used by the systems, almost all systems use linguistic similarity between class names and other of the class description as a basis for determining candidates. Some systems also had applied structural techniques.

In *directory* tests, the Falcon produced the better F–measure, followed by the RiMON, COMA, Prior, and H-Match, respectively. Again, for the *food* tests, the RiMON and Falcon had performed consistently well and achieved the better values of F–measure. H-Match and Prior achieved similar results, followed by COMA. None of the system was able to return the relations of broadMatch and narrowMatch. Similar results were achieved in *conference* tests, where Falcon had achieved better F-measure, followed by RiMON, H-Match, and COMA.

The OAEI 2007 campaign was formed by additional set of tests, namely *environment*, containing thesauri about the environment, and *library*, formed by thesauri about books. For both sets, the relations of broadMatch, narrowMatch and exactMatch were considered. We compared twelve of the evaluated systems: ASMOV, DSSim, Falcon, Lily, OLA, OntoDNA, PriorPlus, RiMON, SAMBO, SEMA, Silas, TaxoMap, and X-SOM. For *benchmark* tests, three systems are relatively ahead (ASMOV, Lily, and RiMON), with three close followers (Falcon, PriorPlus, and OLA).

For *anatomy* tests, systems that use additional background knowledge related to the biomedical domain (SAMBO and ASMOV) clearly generated better alignments compared to the systems that do not use it. The follower were, respectively (values of F–measure): Falcon, TaxoMap, PriorPlus, Lily, X-SOM, and DSSim. However, while Falcon had executed in 12 minutes, SAMBO and ASMOV had used 6 hours and 15 hours, respectively.

In *directory* tests, X-SOM had achieved the highest precision, while the values of F–measure had produced the following order: OLA, PriorPlus, Falcon, Lily, RiMON, ASMOV, DSSim, X-SOM, and OntoDNA. For *food* tests, the Falcon performed best, achieving high precision and recall, followed by RiMON, DSSim, and X-SOM. The fact of the systems returned only exactMatch mappings, significantly had limited recall.

Only Falcon and DSSim had participated in the *environment* task, while in *library* task, Falcon, DSSim and Silas had been evaluated. For both tasks, Falcon had performed better

than the other participants. For *library* task, no participant proposed hierarchical broader and narrower links, which are useful for the applications scenarios at hand. Finally, for *conference* tests, Falcon, OLA, OntoDNA, ASMOV, Lily and SEMA had produced, respectively, the better results.

As the results of OAEI campaigns, Falcon had performed well in the majority of the tests. Specifically for the *benchmark* track, RiMON is in set of better results both in 2006 and 2007. However, similar techniques and strategies, when applied to different ontologies can produce different results. For instance, for *directory* and *anatomy* tracks, OLA and SAMBO performed better than the other participants. Basically, systems that use both ontology and data level information performed well. For complex ontologies, such as the OWL-DL ontologies of the *anatomy*, systems using specific domain knowledge show considerable improvement in the results.

4.2 Negotiation and Argumentation based Ontology Matching

The use of negotiation and argumentation approaches as strategies to combine ontology matching approaches is a promising research area. Different matching techniques can be encapsulated by agents, which negotiate the conflicts that arise from their individual views. Recent proposals for ontology negotiation and argumentation have been proposed in the literature. In this section, such proposals are detailed.

4.2.1 Tamma et al. (2002)

(Tamma et al. [2002]) proposes an negotiation ontology, which serve as the basis for agent negotiation. The idea behind this approach is that negotiation protocols must not be hard-coded in agents, but must be described using a shared ontology of negotiation. The ontology provides the basic vocabulary that an agent and a negotiation host must shared in order to discuss the terms of the participation in the negotiation session. It describes the concept used to describe a negotiation protocol and it is populated by particular protocols – the concepts shared across all possible applications and domains are represented in higher part of the hierarchy, while the concepts in the lowest part of the hierarchy are specific and concern a single negotiation protocol. The ontology itself is not the object being negotiated.

4.2.2 Bailin and Truszkowski (2002)

(Bailin and Truszkowski [2002]) presents an ontology negotiation model which aims to arrive at a common ontology which the agents can use in their particular interaction. It is proposed a protocol that allows agents to discover ontology conflicts (semantic mismatches) and then, through incremental interpretation, clarification, and explanation, establish a common basis for communication with each other.

The *interpretation* determines if a message is properly understood between the agents, where the messages are considered sequences of keywords and the recipient agent tries to interpret each keyword in turn. First, the agent checks its own ontology to see whether the keyword occurs there. If not, the agent queries WordNet to find synonymous of the keyword. Then, it checks the ontology for any of these synonymous. If a synonymous is located in the ontology, it represents an interpretation of the keyword. Considering that there are many synsets for a given keyword, each synonymous must be confirmed by the source of the message – *confirmation of the interpretation.*

When an agent is not able to interpret some of the keywords of a message it has received, it can decide to proceed anyway (if enough other keywords are understood), or it can request a *clarification.* The message source, then, applies the following methods of clarification: (a) locate synonymous in the source agent's ontology; (b) provide a complete set of specialization from the source's ontology (keywords as the union of its subclasses); (c) provide a weak generalization from the source's ontology; or (d) provide a definition in formal logic, defining the keyword as the conjunction of other keywords.

The negotiation process ends with one or both agents modifying their ontology to introduce a new concept. The end result of this process is that each agent converges on a single, shared ontology.

4.2.3 Beun et al. (2004)

(Beun et al. [2004]) proposes a framework for the detection of ontological discrepancies in multiagent systems. The discrepancies can be detected during a communicative situation and the agents can react to these observed discrepancies. Agents may detect discrepancies by type conflicts, ontological gaps, and particular inconsistences that emerge during the conversation process. Depending on the kind od discrepancy, the agent generates a particular feedback message in order to establish alignment of its private ontology with the ontology of the sender.

It is adopted an approach in which agents have a dynamic mental state that contains ontological information about the domain of interest in terms of type theoretical contexts – the agent's ontology are expressed in type theory, which is a logical formalism based on typed

lambda calculus.

The decision criteria for discrepancies is expressed in terms of type theory where the addition of particular information to ontologies yields so called legal or illegal contexts. A legal context is a context where the addition of new information was adhered to the rules of the type system, i.e., the introduction of new predicates in only possible if the type of its arguments is already included in the ontology; otherwise the context is illegal.

In the detection stage of the interpretation process of an incoming message particular information – presuppositions – is extracted from the message. In cases where the addition of presuppositions to the ontology of the receiver yields an illegal context, the receiving agent has to generate particular feedback. Thus, to detect possible ontology discrepancies, the presuppositions are compared with the ontology of the receiver.

Based on a FIPA-compliant agent communication language, it is distinguished between messages to ask and answer questions about the state of the domain and messages for giving feedback at the ontological level. For instance, *query-if*, *confirm*, and *disconfirm* are used to query, confirm and disconfirm that an expression φ is believed to be true of the domain, respectively. The following feedback messages are specified: *query-if*, which is used to ask whether a type theoretical context Γ is part of the recipient's ontology, and *inform*, which is used (according different message's parameters) to report that Γ is new to the sender (i.e., that it is not part of its ontology) or to indicate a mismatch between part Γ of the sender's ontology an part Γ' of the recipient's ontology.

4.2.4 Silva et. al. (2005)

(Silva et al. [2005]) describes an approach for ontology mapping negotiation, where agents are able to achieve consensus about mapping rules defined between two different ontologies. This approach is inserted in the Cooperative Consensus Building module of MAFRA (Maedche et al. [2002]).

The negotiation mechanisms is based on utility functions that evaluate the confidence in a certain mapping rule. According to a confidence value, the mapping rule can be accepted, rejected or negotiated. A mapping is composed of a set of semantic bridges (mapping rules), which describe the semantic relation between a set of entities of the source ontology and a set of entities of the target ontology. Such semantic bridges represent the perspective of an agent on the semantic relations defined between the entities of two ontologies. They are computed by matcher services in MAFRA architecture.

First, each agent performs its own semantic bridging process, generating a mapping. After, the set of semantic bridges are subject of negotiation between both agents. Based on confidences degrees, the semantic bridges can be considered *rejected*, *non-negotiable*, *negotiable*,

proposed, or *mandatory*. The negotiation runs int two phases. The first one intends to build a consensus on mandatory semantic bridges (SB^m), while the second intends to build a consensus on the proposed semantic bridges (SB^p). In the first phase, each agent proposes every $sb^m \in SB^m$ to the other agent. If one sb^m is not accepted by the other agent, the negotiation is closed without a consensus.

In the second phase, each agent proposes every $sb^p \in SB^p$ (not yet negotiated) to the other agent and three situations can occur: (i) the semantic bridge is also proposed by the other agents, thus it is agreed (SB^a); (ii) the semantic bridge is rejected by the other agent, and is therefore rejected; or (iii) the semantic bridge is negotiable by the other agent, therefore categorized as tentatively agreed (SB^t). In the (iii) situation, the semantic bridges are evaluated according to utility functions, representing the overall goal of the negotiation of such semantic bridge.

4.2.5 Diggelen at al. (2006)

(van Diggelen et al. [2006]) proposes a layered communication protocol which incorporates techniques for ontology exchange, where the agents gradually build towards a semantically integrated system by establishing shared ontologies. The agents agree on a common ontology in a decentralized way, i.e., every agent increments its own ontology with the necessary concepts.

The protocol is composed by three layers: normal communication protocol (NCP), concept definition protocol (CDP), and concept explication protocol (CEP). NCP deals with normal agent communication, i.e., the kind of social interaction which agents normally exhibit when no ontology problems exist in the system, while the other two layers are added to the protocol to deal with ontology problems.

The upper layer NCP deals with message composition and interpretation, and the decision of when to switch to CDP. It is proposed for its layer a simple communication mechanism that exploits some features of partially shared ontologies, namely the distinction between native concepts (i.e., concepts in the original agent's ontology) and acquired concepts (i.e., learnt concepts during the negotiation), and equivalence mappings. The message composition involves translating the native concept the sender intents to convey to an equivalent shared concept (i.e., shared concepts is common between the two agents) and use this in a message. Message interpretation involves translating the concept used in the message to an equivalent native concept. The agent switches to CDP when no equivalent shared concept is available in its ontology. Moreover, once an agent has used an unknowingly shared concept (i.e., both agents know the concept but do not know this of each other) in the message, this concept becomes shared.

In CDP, the agent tries to convey the meaning of a concept by stating the relations with other

concepts. If the definitions of other concepts enable the agent to derive a complete meaning of the concept, then the agent switches back to NCP. An agent considers the meaning of an acquired concept complete if it knows the relation with every other concept in the ontology. If there are not sufficient shared concepts available to convey the complete meaning, the agent switches to CEP.

Finally, CEP aims to convey the meaning of a concept when no satisfactory definition of the concept in terms of other concepts can be given. It is used a concept classifier to classify concept instances. The agent (sender), upon explicating a concept, communicates a number of positive and a number of negative examples of the concepts. The receiver classifies these examples using the concept classifiers from its own ontology. After the concept classification, the receiver switches to CDP.

4.2.6 Laera et al. (2006,2007)

(Laera et al. [2006] and Laera et al. [2007]) propose to use an argument framework to deal with arguments that support or oppose candidate mappings, according to agent's preference. The argumentation framework relies on a formal argumentation schema and on an encoding of the agents' preferences between particular kinds of arguments, distinguishing between alignments valid for all agents and those specific to a particular agent. The Value-based Argumentation Framework is used, where audiences represent different preference between the categories of arguments that are identified in the context of ontology matching. Each agent has a (partial or total) pre-ordering of preferences over different types of ontology mismatches (Pref).

The alignments are generated by a dedicated agent, called an Ontology Alignment Service (OAS). Each alignment is a set of all possible mappings between the two ontologies. A mapping is a tuple m = (e, e', n, R), where e and e' are the entities (concepts, relations or individuals) between which a relation is asserted by the correspondence; n is a degree of confidence in that correspondence; and R is the relation (e.g., equivalence, more general, etc.) holding between e and e'. For each correspondence m, an OAS is able to provide a set of justifications G, that explain why it has generated a candidate mapping. The agents use such information to exchange arguments supplying the reasons for their mapping choices.

Every agent has a private threshold value ϵ which is used to compare to the degree of confidence that an OAS associates with each mapping. This threshold together the pre-ordering of preferences are used used to generate the arguments for and against a mapping.

An argument x is a triple x = (h,G,m), where m is a mapping m, G is the grounds justifying a prima facie belief that the correspondence does, or does not hold; h is one of +, - depending on whether the argument is that m does or does not hold. Counter-arguments are generated by different values of h, when a mapping m refers to the same entities e and e'. The grounds

justifying *G* correspondences are classified according the following groups:

- semantic (M): the sets of models of two entities do or do not compare;
- internal structural (IS): two entities share more or less internal structure – e.g., the value range or cardinality of their attributes;
- external structural (ES): the set of relations, each of two entities have, with other entities do or do not compare;
- terminological (T): the names of two entities share more or less lexical features;
- extensional (E): the known extension of two entities do or do not compare;

These classes are used as types for the values V $\in$ VAF (i.e., V = M, IS,ES,T,E). For instance, an audience may indicate that terminological arguments are preferred to semantic arguments. This preference is determined according to the kind of ontology. The pre-ordering of preferences Pref for each agent is over V, corresponding to the specification of an audience. Specifically, for each candidate mapping m, if there exist justification(s) G for m that corresponds to the highest preferences Pref (with the respect of the pre-ordering), assuming n is greater than its private threshold ϵ", an agent will generate arguments x = (G,m,+). If not, the agent will generate arguments against: x = (G, m,-).

The argumentation process takes four main steps: (i) for each agent, for each candidate mapping is constructed an argumentation framework, by specifying the set of arguments (according agents's preferences and threshold) and by determining the attacks between them; (ii) the individuals frameworks are merged, by forming the union of the individual argument sets and attack relations of the multiple agents, and then extend the attack relations by computing the attacks between the arguments present in the framework of each agent with the arguments of all the other agents; (iii) for each VAF, is is determined which of the arguments are undefeated by attacks from other arguments, given a value ordering – the global view is considered by taking the union of these preferred extensions for each audience; and (iv) the arguments in every preferred extension of every audience are considered – the mappings that have only arguments for are included in the a set called *agreed alignments*, the mappings that have only arguments against are rejected, and the mappings which are in some preferred extension of every audience are part of the set called *agreeable alignments*.

4.2.7 Comparison of the Negotiation and Argumentation based Systems

Most of the ontology negotiation-based systems aims to arrive at a common ontology which the agents can use in their interactions, as proposed by (Bailin and Truszkowski [2002]),

(Beun et al. [2004]), and (van Diggelen et al. [2006]). Differently from these proposals, (Tamma et al. [2002]) presents ontology which describe the basic concepts of a negotiation process. Such ontology is not the object being negotiated.

While (Bailin and Truszkowski [2002]) proposed a protocol where incremental interpretation, clarification, and explanation steps are used to establish a common basis for communication between the agents; (Beun et al. [2004]) defines sets of discrepancies and feedbacks to solve ontological discrepancies; and (van Diggelen et al. [2006]) presents a layered communication protocol, where each layer is able to solve a kind of ontological mismatch. In such systems, the agent ontologies are incrementally changed during the negotiation step.

On the other perspective, (Silva et al. [2005]) describes an ontology mapping negotiation based on utility functions, where agents are able to achieve consensus about mapping rules defined between two different ontologies. Each agent keep our ontologies unaltered. However, the mappings are specified by services outside the negotiation framework. However, the system is highly dependent on the MAFRA framework and cannot be flexibly applied in other environments.

The use of argumentation in ontology matching is proposed only by (Laera et al. [2006], Laera et al. [2007]), in order to deal with arguments that support or oppose candidate correspondences between ontologies. However, the candidate mappings are obtained from an Ontology Mapping Repository (OMR) – the focus is not how the mappings are computed – and argumentation is used to accommodate different agents' preferences. In such approach, the agents keep their own ontologies unaltered.

Basically, the closer proposals to our are from (Silva et al. [2005]) and (Laera et al. [2006], Laera et al. [2007]). However, in this thesis, the mappings are computed by specialized agents, which negotiate or argument to solve conflicts between the individual results.

4.3 Summary

This chapter presented a review of the main matching systems proposed in the literature. It was not an exhaustive but a representative overview of the matching systems. The systems were classified into two broad categories: *standard systems*, which does not make use of negotiation or argumentation approaches, and *negotiation and argumentation based* systems.

The standard systems were grouped according to the kind of input data they use in the matching process, namely into the categories *ontology level*, where 24 representative systems were described, *data level*, with a description of 11 systems, and *ontology and data levels*, where 16 systems were presented. Basically, most of such systems are based on sequential aggregation of matching techniques, and statistical or machine learning approaches, respectively. Few systems use some form of reasoning (as one of the major advantages of OWL)

in order to take the meaning of the ontologies into account. Other important issue is related with the kind of semantic relations that the systems are able to deal. Few systems return broader and narrower relations, which are useful for the applications.

When compared with the number of *standard* systems, few approaches are proposed in the fields of negotiation and argumentation ontology matching. Most of such systems had focused on the construction of a common shared ontology as a first step to solve the semantic heterogeneity in agent communication situations. The closer proposal than the one presented in this thesis was proposed by Laera and colleagues, where an argument framework is used to deal with arguments that support or oppose candidate correspondences between ontologies. However, the candidate mappings are obtained from an Ontology Mapping Repository (OMR) – the focus is not how the mappings are computed – and argumentation is used to accommodate different agent's preferences. In the approach proposed in this thesis, the mappings are computed by specialized agents which negotiate and argument to solve conflicts between their individually obtained results, as detailed in the following.

The next chapter present the whole proposed agent-based model. The matcher agents, the negotiation model, and the argumentation framework are detailed.

Chapter 5

Composite Approaches for Ontology Matching

Different matchers working on the basis of particular approaches arrive to distinct matching results that must be shared, compared, chosen and agreed. In order to deal with this problem, we propose an agent-based approach where cooperative agents try to agree on a final matching result. Two agent-based proposals are presented: *negotiation-based* and *argumentation-based*. In both proposals, cooperative agents apply individual matching algorithms and cooperate in order to exchange their local results. In the *negotiation-based* approach, a protocol based on voting is used, where each vote represents the view of an agent on the matching between entities of the ontologies. On the other hand, the *argumentation-based* approach is based on the interaction of different arguments representing matching results. Such arguments are evaluated following a notion of acceptability, in order to select the valid arguments.

The term "agent" is used in this thesis in order to following the terminology of the automated negotiation and argumentation literature. However, the proprieties of agents (i.e., full autonomy, learning, and reasoning), as explored in the agent literature, are not fully considered. In the context of this thesis, "agent" means an entity that apply a specific technique to solve a problem, and communicate with other entities to exchange their points of view.

This chapter presents the proposed *cooperative negotiation* and *argumentation* models. The matcher agents that represent the individual matching techniques are also described. These techniques are variations of the standard ones presented in the Section 3.2, in order mainly to support ontology entities composed by multi-words.

The chapter is organized as follows. Section 5.1 presents the matcher agents. Section 5.2 presents the negotiation-based approach. The organization of the society of matcher agents and the negotiation protocol are detailed. Section 5.3 presents the argumentation-

based approach. An extension of the Value Argumentation Framework (VAF) (Bench-Capon [2003]), in order to represent arguments with confidence degrees, is presented and the argumentation process is detailed. Section 5.4 comments on the comparison between the two approaches. Finally, Section 5.5 presents the summary of the chapter in the context of the thesis.

5.1 Matcher Agents

The approaches for ontology matching can be grouped into three representative categories: *syntactic*, *semantic*, and *structural*. These categories are used as basis to the definition of the matcher agents proposed in this thesis.

The three matchers are specified to deal with composite terms and different semantic relations, namely *exactMatch*, *narrowMatch*, *broadMatch*, and *relatedMatch*. As commented in the previous chapter, few systems are able to deal with such relations (for instance, S-Match, TaxoMap, and oMap). The majority of the systems output a confidence measure in range [0,1] for equivalence relations. For real world applications, such as *library* and *anatomy* tasks in the OAEI, hierarchical broader and narrower links are useful.

5.1.1 Syntactic matcher

The syntactic matcher considers labels of the ontology entities as a sequence of characters. It is based on the Levenshtein distance (Levenshtein [1966]) and considers the length of the compared labels to compute syntactic similarity (Maedche and Staab [2002]). The syntactic similarity (SS) between two entities e_s and e_t is:

$$SS(e_s, e_t) = max\left(0, \frac{min(\mid e_s \mid, \mid e_t \mid) - ED(e_s, e_t))}{min(\mid e_s \mid, \mid e_t \mid)}\right)$$

First, the labels are parsed into tokens and the syntactic similarity is computed for each token. Most matchers (basically all that use edit distance measure) consider composite terms as a unique label in order to compute the distance between them. As a results, labels as for instance "camera-and-photo" and "photo-and-camera" are not correctly matched.

Differently of the most syntactic approaches, which output only the *equivalence* relation, the syntactic matcher considers four relations: *exactMatch*, *broadMatch*, *narrowMatch* and *relatedMatch*. Consider two entities from the source and target ontologies, e_s and e_t, where $e_s = a_{s1}, ..., a_{si}$ and $e_t = a_{t1}, ..., a_{tj}$. i is the number of tokens (atoms) for e_s, and j is the number of tokens for e_t. $matches(e_s, e_t)$ indicates the number of tokens that syntactically

match between e_s and e_t — two tokens match if the *edit distance* is greater than a given threshold. The mapping relation between e_s and e_t is defined as follows (where the strength is $matches(e_s, e_t)/max(i, j)$):

- $exactMatch(e_s, e_t)$ if $matches(e_s, e_t) = i = j$;
- $narrowMatch(e_s, e_t)$ if $i > j$ and $matches(e_s, e_t) = j$;
- $broadMatch(e_s, e_t)$ if $i < j$ and $matches(e_s, e_t) = i$;
- $relatedMatch(e_s, e_t)$ in all the other cases where $matches(e_s, e_t) > 0$.

For instance, for the entities e_s = "photo-camera" and e_t = "camera-photo", an *exactMatch*(e_s,e_t) with strength 1 is obtained; and for the entities e_s = "science" and e_t = "computer-science", a *broadMatch*(e_s,e_t) with strength 0.5 is returned, where $matches(e_s, e_t)$ = 1 and $max(i, j)$ = 2.

5.1.2 Semantic matcher

The semantic matcher is based on semantic relations (i.e., synonym, hyponym, and hypernym) between entities to measure the similarity between them, on the basis of a thesauri. Differently of systems such as XClust, S-Match, and DSSim, which consider only synonymous relations, the semantic matcher retrieves also hyponym and hypernym relations, looking for tokens into composite terms.

Consider e_s and e_t, where e_s = a_{s1}, ..., a_{si} and e_t = a_{t1}, ..., a_{tj}. $matches(e_s, e_t)$ indicates the number of tokens that are *synonymous* between e_s and e_t. $rel(e_t, e_s)$ indicates the semantic relation between e_s and e_t, when searching them in the thesauri without decomposing the labels of the entities into tokens.

The mapping relation between e_s and e_t is defined as follows (where the strength is given by $matches(e_s, e_t)/max(i, j)$ or 1 in the cases where $rel(e_s, e_t) \neq \emptyset$):

- $exactMatch(e_s, e_t)$ if $rel(e_s, e_t)$ = synonymous or $matches(e_s, e_t) = i = j$;
- $narrowMatch(e_s, e_t)$ if $rel(e_s, e_t)$ = hyponym or ($i > j$ and $matches(e_s, e_t) = j$);
- $broadMatch(e_s, e_t)$ if $rel(e_s, e_t)$ = hypernym or ($i < j$ and $matches(e_s, e_t) = i$);
- $relatedMatch(e_s, e_t)$ in all the other cases where $matches(e_s, e_t) > 0$.

For instance, for the entities e_s = "photo-camera" and e_t = "camera-photo", a thesauri as WordNet is not able to return the synsets between these multi-words labeled entities. However, decomposing the entities "camera" is synonymous of "camera" and "photo" is synonymous of "photo". So, an *exactMatch*(e_s,e_t) with strength 1 is obtained. For the entities e_s = "personal-computer" and e_t = "pc", WordNet contains a direct relation of synonymous in these synsets, i.e., an *exactMatch*(e_s,e_t) with strength 1 is obtained.

5.1.3 Structural matcher

The structural matcher is a taxonomy-based matcher that considers the positions of the terms in the ontology hierarchy to verify if the entities must be matched. The super-classes of the two ontology entities are taking into account in the matching process. Most structural matchers consider positions of the terms in the hierarchy or the number of sub-classes. The proposed structural matcher looks inside the labels of each super-class, considering the intensional semantic of a concept based on the composition of the label describing the concept.

It is based on the *taxonomy overlap* (Maedche and Staab [2002])and *semantic cotopy* (Maedche and Staab [2002]) measures. The intensional semantics of a concept c_i may be seen to be constituted by the *semantic cotopy* (SC) of c_i, i.e., all its super and sub concepts: SC(c_i,O_i) = $\{c_j \in C_i \mid c_i \leq_C c_j \vee c_j \leq_C c_i\}$. Based on the SC, the *taxonomy overlap* (TO) is defined as:

$$TO(ci, O_s, O_t) = \frac{\mid SC(c_i, O_s) \cap SC(c_i, O_t) \mid}{\mid SC(c_i, O_s) \cup SC(c_i, O_t) \mid}$$

In order to consider only the super concepts of the entities being compared, the intensional semantics of c_i is adapted to be constituted of the *n* super concepts of c_i: $SC_n(c_i,O_i) = \{c_j \in C_i \mid c_i \leq_C c_j\}$, with.

$$TO_n(c_i, O_s, O_t) = \frac{\mid SC_n(c_i, O_s) \cap SC_n(c_i, O_t) \mid}{\mid SC_n(c_i, O_s) \cup SC_n(c_i, O_t) \mid}$$

The mapping relation between e_s and e_t is defined as follows:

- $exactMatch(e_s, e_t)$ if $TO_n(e_s, e_t) = 1$
- $broadMatch(e_s, e_t)$ if $e_s \in SC_n(e_t, O_s, O_t) - e_t$
- $narrowMatch(e_s, e_t)$ if $e_t \in SC_n(e_s, O_s, O_t) - e_s$
- $relatedMatch(e_s, e_t)$ if $\mid (SC_n(e_s, O_s, O_t) - e_s) \cap (SC_n(e_t, O_s, O_t) - e_t) \mid > 0$

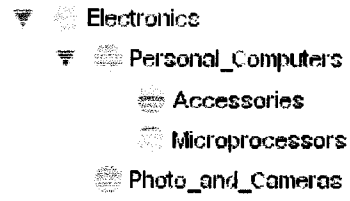

Figure 5.1: Source ontology.

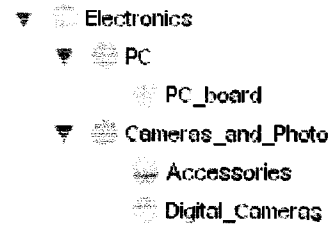

Figure 5.2: Target ontology.

In order to compute the SC_n and TO_n, the syntactic or semantic similarity can be used.

Consider the ontologies source (Figure 5.1) and target (Figure 5.2)[1], and the matching between the entities "personal-computer" and "pc":

- SC_n("PersonalComputer",O_s) = {Electronic, PersonalComputer}
- SC_n("PC",O_t) = {Electronic, PC}

$$TO_n(e_s, e_t, O_s, O_t) = \frac{| \, Electronic, PersonalComputer \cap Electronic, PC \, |}{| \, Electronic, PersonalComputer \cup Electronic, PC \, |}$$

By using the semantic similarity, $TO_n = 2 \setminus 2 = 1$, and then the entities "personal-computer" and "pc" are matched via an *exactMatch* relation, with strength 1.

When considering the matching between the entities e_s = "electronics" and e_t = "pc":

- SC_n("Electronic",O_s) = {Electronic}
- SC_n("PC",O_t) = {Electronic, PC}

$$TO_n(e_s, e_t, O_s, O_t) = \frac{| \, Electronic \cap Electronic, PC \, |}{| \, Electronic \cup Electronic, PC \, |}$$

[1]Ontologies available in http://dit.unitn.it/ accord/Experimentaldesign.html(Test 4)

In this case, $TO_n = 1 \setminus 2 = 0.5 \neq 1$, but "electronic" $\in SC_n(e_t, O_s, O_t) - e_t$ = {Electronic, PC} - PC. Then, the entities are related by a relation of ***broadMatch***, with strength 0.5.

5.2 Cooperative Negotiation for Ontology Matching

The idea behind the cooperative negotiation is that agents sharing common objectives try to arrive at mutual agreement regarding their partial (and sometimes different) views of the problem they are trying to solve. This idea can be adapted to the ontology matching problem. Different matching approaches suggest to consider different perspectives of the matching problem. The conflicts that arise from the different perspectives can be solved by negotiation, where the matching techniques are encapsulated in so called agents.

In the proposed negotiation model (Trojahn et al. [2006], Trojahn et al.), the agents apply different matching techniques – syntactic, semantic and structural – to match entities of two different ontologies. The distinct mapping results are shared, compared, chosen and agreed, and a final mapping result is obtained. Following, the organization of the agent society is presented.

5.2.1 Organization of the agent society

The negotiation model is described according to an agent society (Figure 5.3), using the Moise+ model Hübner et al. [2002]. This model proposes three dimensions for the organization of agent societies: structural, functional and deontic. The structural dimension defines what agents could do in their environment (their roles). The functional dimension defines how agents execute their goals. The deontic dimension defines the permissions and obligations of a role in a goal.

According to Hübner et al. [2002] and Hubner [2003], structural specification has three main concepts, roles, role relations and groups that are used to build, respectively, the individual, social and collective structural levels of an organization. The individual level is composed by the roles of the organization. A role means a set of constraints that an agent ought to follow when it accepts to play that role in a group. The following roles are identified in the proposed organization:

- Mediator: this role is responsible for mediating the negotiation process, sending and receiving messages to and from the matcher agents.
- Matcher: this role is responsible for giving an output between two ontology mappings (i.e., encapsulates the matcher algorithms). One matcher could assume the syntactic, semantic or structural role.

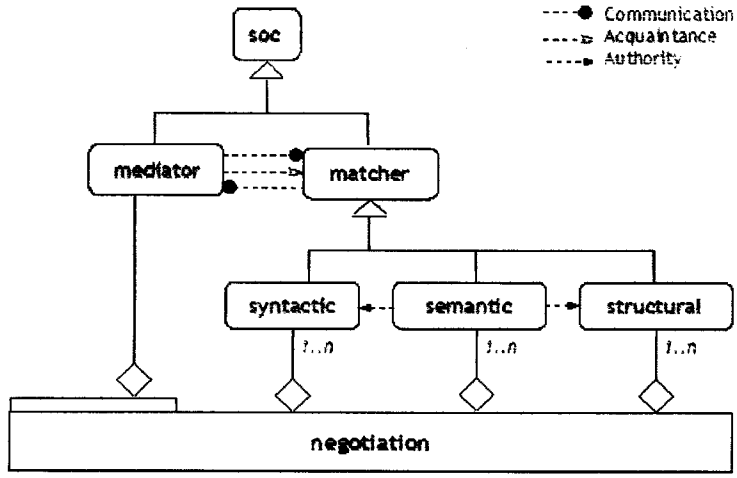

Figure 5.3: Negotiation organizational model.

In the social level are defined the kinds of relations among roles that directly constrain the agents. Some of the possible relations are:

- Acquaintance (acq): agents playing a source role are allowed to have a representation of the agents playing the destination role. In Figure 5.3, this kind of relation is present between the source role mediator and the destination role matcher.
- Communication (com): agents playing a source role are allowed to communicate with agents that play the destination role. In Figure 5.3 this kind of relation is present between the source role mediator and the destination role matcher (by heritage, specific matchers).
- Authority (aut): agents playing a source role has authority upon agent playing destination role. In Figure 5.3 this kind of relation is present between the source role matcher semantic and the destination roles matchers syntactic and structural.

The collective level specifies the group formation inside the organization. A group is composed by the roles that the system could assume, the sub-groups that could be created inside a group, the links (relations) valid for agent and by the cardinality. A group can have intra-groups links and inter-groups links. The intra-group links state that an agent playing the link source role in a group is linked to all agents playing the destination role in the same group or in its sub-groups. The inter-group links state that an agent playing the source role is linked to all agents playing the destination role despite the groups these agents belong to Hübner et al. [2002]. Links intra-group are represented by a hatched line and links inter-groups are represented by a continue line. This specification defines only a group called negotiation and all links are intra-group.

Based on the structural specification of the proposed organization, the society is composed by one agent that assumes the mediator role and three agents that assume the matcher role.

5.2.2 Negotiation process

Basically, the negotiation process involves two phases. First, the agents work in an independent manner, applying a specific matching approach and generating a set of negotiation objects. A negotiation object is a 2-tuple (A, m), where A indicates the matcher generating the mapping m. A mapping m is a 5-tuple (e,e',h,R,s), where e and e' are ontology entities; h is a value of +,- indicating if the mapping between e and e' hold or not; R is the relation between e and e'; and s is a continuous values indicating the *degree of confidence* in the mapping, which can be used to filter the mappings below a *threshold*. Second, the set of negotiation objects is negotiated among the agents. The negotiation process involves one mediator and several matcher agents.

Figure 5.4 shows an AUML (Agent Unified Modeling Language) interaction diagram with the messages changed between the agents during a negotiation round. It is used an extension of AUML-2 standard to represent agents' actions (the actions are placed centered over the lifeline of the named agent). The interaction diagram refers to negotiation of the mapping between the entities e = "personal computer" and e' = "pc" (Figures 5.1 and 5.2, respectively). Three matchers are considered, syntactic, semantic, and structural, which return the following mappings, respectively: $m_{syntactic}$ = ("personal-computer","pc",-,exactMatch,1.0), $m_{semantic}$ = ("personal-computer","pc",+,exactMatch,1.0), and $m_{structural}$ = ("personal- computer", "pc",+,exactMatch,1.0).

The negotiation process starts with the mediator agent asking to the matcher agents for its number of "mappings". The first matcher agent to generate a proposal is one that has the greatest number of "mappings" (syntactic agent, in the specific example).

The proposal contains the first negotiation object that still wasn't evaluated by the agent. This proposal is then sent to the mediator agent, which sends it to other agents (in the specific example, the syntactic agent proposes a negative mapping (h = -) to the mapping between the entities "personal computer" and "pc"). Each agent then evaluates the proposal, searching for an equivalent negotiation object. One negotiation object is equivalent to another when both refers to same entities e and e' in the two ontologies.

If an equivalent negotiation object has the same value of h, the agent accepts the proposal. Otherwise, if the agent has a different value of h in the negotiation object, its object negotiation is sent as a counter-proposal to the mediator agent, which evaluates the several counter-proposals received (several agents can send a counter-proposal). In the example, semantic and structural agents have generated counter-proposals, indicating a positive mapping (h = +) between the compared entities.

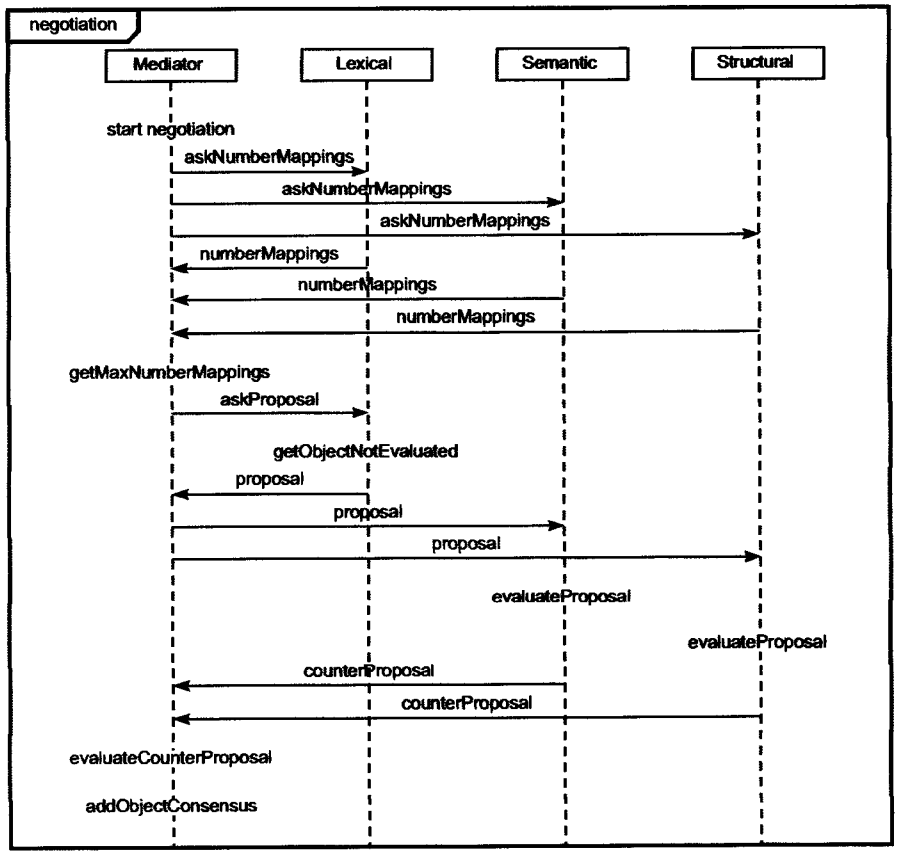

Figure 5.4: AUML negotiation interaction.

The mediator selects one counter-proposal that has the greater number of hits. If the two kinds of matching, negative and positive, receive equals number of hits, the mediator uses a global preference order (for instance, semantic > structural > syntactic) to select the final result. When a proposal is accepted by all agents or a counter-proposal consensus is obtained, the mediator adds the corresponding negotiation object in a consensus negotiation set and the matcher agents mark its equivalent one as evaluated. The negotiation ends when all negotiation objects are evaluated.

Moreover, when two matchers indicate different matching relations for the corresponding entities, the relation indicated by the matcher with greater preference in the preference order is selected.

5.2.3 Negotiation algorithm

Table 5.1 presents the negotiation algorithm to define the matching consensus among the matcher agents. It describes to a round of negotiation, where a negotiation object (matching between two entities) is evaluated. The negotiation ends when all negotiation objects have been evaluated. A proposal P refers to a negotiation object. A consensus C is returned by the algorithm.

In order to simplify the algorithm description, the specification of the selection of the relation that hold between the entities when different ones are counter proposed is not detailed. In this case, the relation with greater number of hints is selected and in the cases where two relations receive the same number of votes, the relation indicated by the agent with greater preference is selected.

5.3 Argumentation Framework for Ontology Matching

Following the line of approaches agent-based, argumentation is an innovative and promising option, where arguments in favor or against mappings between ontology entities are represented and processed. In these approaches, matchers encapsulating different categories of matching approaches, generate a set of arguments that represent the mappings between the ontology entities. According to the definition of attacking relations, an argument for a mapping generated by one matcher can be supported or attacked by other arguments from other matchers. Based on the argumentation framework instantiation (using specific attacking relation and preference order between the arguments), the matchers compute their preferred set of arguments. The arguments in such preferred sets are viewed as the set of globally acceptable arguments (mappings).

The argumentation framework proposed in the context of this thesis – Strength-based Ar-

Table 5.1: Algorithm for defining consensus via negotiation.

Require: mediator *M*, matchers *Ag*, mappings *Map*, preference order *Pref*
Output: consensus *C*

```
1: (M)      for each Ag_i ∈ Ag
                askNumberMappings
            end for
2: (Ag_1..n) sendNumberMappings(nM)
3: (M)      selectAgStartNeg(Ag_i)
4: (M)      askProposal(Ag_i)
5: (Ag_i)   for each Map_i,j ∈ Map_i
                P = getMapNotEvaluated()
                sendProposal(P)
            end for
6: (M)      for each Ag_k ∈ where k ≠ i
                sendProposal(P)
            end for
7: (Ag_k)   for each Map_k,j
                if (e_k,j = e_P) and (e'_k,j = e'_P) and (h_k,j = h_P) then
                    acceptProposal(P)
                else
                    sendCounterProposal(P')
                end if
            end for
8: (M)      if all Ag_k accept(P) then
                for each P
                    if Rel_k ≠ Rel_P then
                        Rel_P = Rel_i | Rel_i ∈ P_i and Ag_i ≫_Pref Ag_k
                    end if
                end for
                add(P,C)
            else
                for each P'_k
                    positive = countH+(P'_k)
                    negative = countH-(P'_k)
                end for
                if (negative > positive) then
                    h_P' = -
                else
                    if (positive > negative) then
                        h_P' = +
                    else
                        h_P' = h_k | h_k ∈ P_k and Ag_k ≫_Pref Ag_j
                    end if
                end if
                add(P',C)
            end if
```

gumentation Framework (S-VAF) – is based on the Value-based Argumentation Framework (VAF) (Bench-Capon [2003]). The VAF is extended in order to represent arguments with confidence degrees. Originally, the VAF allows to determine which arguments are acceptable, with respect to the different *audiences*, which can characterize different preferences between agents. However, the notion of acceptability of arguments is based on the preferences according to the audiences. The quality of the argument itself is not take into account. Regarding other argumentation frameworks, such as the Preference Argumentation Framework (PAF) (Amgoud and Cayrol [1998]), it considers the quality of the arguments without consider the context (audience) in that arguments are inserted.

In order to define a notion of acceptability based both in the quality of the argument and the context that it occurs, the VAF is extended with confidence degree representing the quality of the arguments being associated to each argument. An audience represents a preference order between different matchers (for instance, semantic matcher is preferred to the syntactic matcher). The idea behind the S-VAF is to arrive at a set of arguments that are acceptable for all audiences. i.e., the set of mappings that are seen as correct for all matchers.

In the next section, the proposed argumentation framework is detailed.

5.3.1 Strength-based Argumentation Framework (S-VAF)

The *strength* represents the confidence that an agent has in some argument. One element has been added to VAF: a function which maps from arguments to real values from the interval [0,1]. Such measure is a relevant criterion in the ontology matching domain. Many matching tools output mappings with a strength that reflects the confidence they have in the similarity of the entities involved in the correspondence. These confidence levels are usually derived from the similarity assessments made during the ontology matching process, *e.g.* from an edit distance measure between labels, or an overlap measure between instance sets.

In previous work (Trojahn et al. [2008a], Trojahn et al. [2008d], Trojahn et al. [2008c]), discrete categories to represent the strength of the arguments – *certainty* and *uncertainty* were used. The use of continuous strength has two main advantages: (i) the set of mappings can be reduced by filtering them using continuous thresholds, which can be adjusted by the user; (ii) continuous values are more representative to indicate the degree de similarity between the entities (specially when such measure considers, for instance, the analysis of composite labeling entities).

In a S-VAF, it is important to distinguish the difference between *values* and *strengths*. There are different types of agents representing different matching approaches. Each approach represent a *value* and each agent represents an *audience*, with preferences between the *values*. The *values* are used to determine the preference between the different agents. Moreover, each agent generates arguments with a *strength*, based on the confidence returned

by the matching technique. So, we extended the VAF in order to define a new notion of argument acceptability which combines *values* (related with the agent's preference) and *strength* (confidence degree of an argument). If our criterion was based only on the *strength* of the arguments, a Preference Based Argumentation Framework could be used (Amgoud and Cayrol [1998]).

A S-VAF has the following definitions:

Definition 1 A Strength based Argumentation Framework (S-VAF) is a 6-tuple *(AR, attacks,V,val,P,valS)* where *(AR,attacks,V,val,P)* is a value-based argumentation framework, and *valS* is a function which maps from elements of *AR* to real values from the interval [0,1] representing the *strength* of the argument.

Definition 2 An argument $x \in AR$ defeats$_a$ (or *successfully attacks*) an argument $y \in AR$ for audience a if and only if $attacks(x,y) \wedge ((valS(x) > valS(y)) \vee (\neg\ valpref(val(y),val(x)) \wedge (\neg\ (valS(y) > valS(x)))))$.

An attack succeeds if (a) the *strength* of the attacking argument is greater than the *strength* of the argument being attacked; or if (b) the argument being attacked does not have greater preference value than attacking argument (or if both arguments relate to the same preference values) and the *strength* of the argument being attacked is not greater than the attacking argument.

Figure 5.5 shows two S-VAFs with their corresponding set of arguments regarding three audiences: *syntactic*, *semantic*, and *structural*. Such S-VAFs represent the matching between the entities "personal computer" and "pc". In the setting (a), the syntactic matcher outputs an argument against the mapping, with strength 0.5; both semantic and structural matchers output arguments in favor of the mapping, with strength 1.0. In the setting (b) the syntactic matcher outputs an argument with strength 1.0.

Regarding the notion of *successfully attacks*, in the setting (a), the argument A is successfully attacked by the arguments B and C – item (a) of the *Definition 2* – independently of the syntactic' preference order. However, when considering the argument A with strength 1.0, the arguments B and C do not successfully attack it, i.e., when the arguments relate the same strength, the preference order is taking into account – item (b) of the *Definition 2*.

Definition 3 An argument $A \in AR$ is *acceptable* to audience a ($acceptable_a$) with respect to set of arguments S, $acceptable_a(A,S)$ if $(\forall x)\ ((x \in AR\ \&\ defeats_a\ (x,A)) \longrightarrow (\exists y)((y \in S)\ \&\ defeats_a(y,x)))$.

Regarding the S-VAFs of Figure 5.5, in the setting (a), for the audience syntactic, the argument A is not acceptable because it is successfully attacked by B and C, which are not

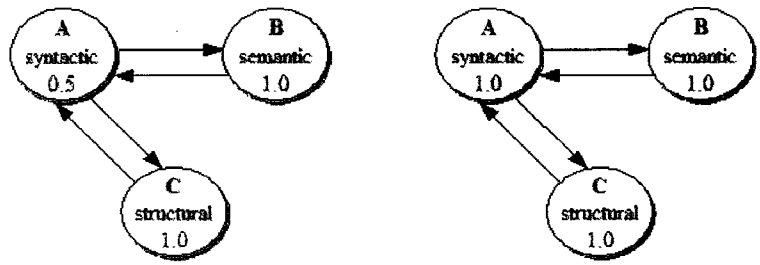

Figure 5.5: S-VAFs examples.

attacked by other arguments. The arguments B and C are acceptable. For the audiences semantic and structural, the arguments B and C are acceptable (A is weaker than B and C). In the setting (b), for the audience syntactic, A successfully attacks the arguments B and C, which are not defeated. So, the argument A is acceptable. As in the setting (a), for the audiences semantic and structural, the arguments B and C are acceptable (A has the same strength than B and C but it is not preferred regarding the audiences).

Definition 4 A set S of arguments is *conflict-free* for audience a if $(\forall x)(\forall y)\ ((x \in S \wedge y \in S) \longrightarrow (\neg attacks(x, y) \vee (\neg(valS(x) > valS(y)) \wedge (valpref(val(y), val(x)) (\vee (valS(y) > valS(x))))))$.

Definition 5 A set of argument S in the S-VAF is a *preferred extension* for audience a ($preferred_a$) if it is a maximal (with respect to set inclusion) *admissible* for audience a of AR.

Definition 6 An argument $x \in AR$ is *subjectively* acceptable if and only if x appears in the preferred extension for some specific audiences but not all. An argument $x \in AR$ is *objectively* acceptable if and only if, x appears in the preferred extension for every specific audience. An argument which is neither objectively nor subjectively acceptable is said to be *indefensible*.

For the setting (a), the preferred extensions are: syntactic matcher = {B} – B attacks A and then A is removed and in the second iteration of argumentation process, C does not attacks B, as detailed in the Section 5.3.3; semantic matcher = {B}; and structural matcher = {C}. In the setting (b), the preferred extensions are: syntactic matcher = {A}; semantic matcher = {B}; and structural matcher = {C}. In both settings, the arguments A, B, and C are *subjectively acceptable*. However, in the setting (a) there is a consensus in the sense that the arguments B and C are in favor of the mapping.

Based on this example, if *objectively acceptable* arguments refer to the same kind of mapping (positive or negative), such arguments are considered as a *global consensus*, i.e., the

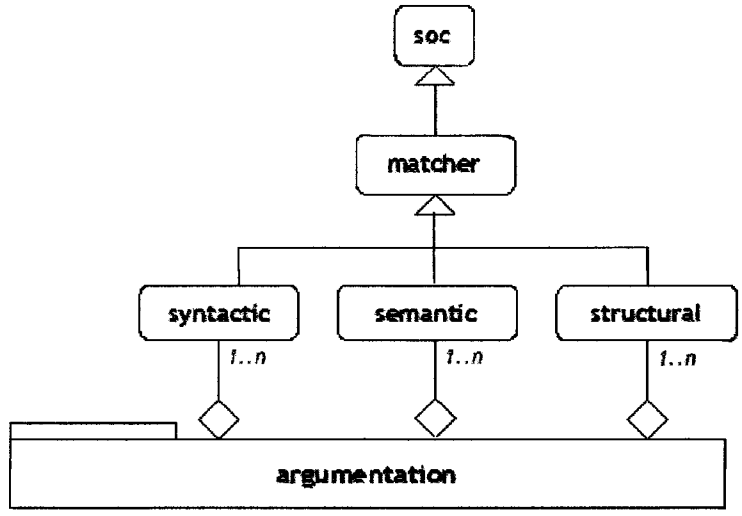

Figure 5.6: Argumentation organizational model.

arguments that have the same value of h in all preferred extensions. On the other hand, *local consensus* refers the set of mappings that are in some preferred extension. In the example above, for the setting (a) the arguments B and C refers to mappings that must be in the *global consensus* set. It leads to the definition of *global consensus* and *local consensus*:

Definition 7 An argument $x \in AR$ is in the *global consensus* if and only if x appears in the preferred extension for every specific audience or if the mapping $m \in x$ has the same h in every audience preferred extension. An argument $x \in AR$ is in the *local consensus* if and only if x appears in the preferred extension for some specific audience or if the mapping $m \in x$ has the same h in some audience preferred extension. An argument which is neither in the global or local consensus is said to be *indefensible*.

5.3.2 Organization of the agents society

The Moise+ model is used to describe the argumentation model (Figure 5.6). In this society, only the matcher role is identified, which is responsible for giving an output between two ontology matching (i.e., encapsulate the matching algorithms). Differently from the negotiation model, there is no role responsible for mediating the argumentation process. The possible relation between the agents is the communication, where the agents playing a source role are allowed to communicate with agents playing the destination role. This kind of relation is present through the communication among the three matcher agents within an agent society.

5.3.3 Argumentation Process

In a S-VAF, the values $v \in V$ represent different matching approaches used by the agents. Three approaches are considered: syntactic (L), semantic (S), and structural (E), then $V = \{L,S,E\}$. Each audience has an ordering preference between the *values*. For instance, the syntactic agent represents an audience where the value L is preferred to the values S and E. The idea is not to have an individual audience with preference between the agents (i.e., semantic agent is preferred to the other agents), but it is to try accommodate different audiences (agents) and their preferences. The idea is to obtain a consensus when using different matching techniques, which are represented by different preference between values.

The argumentation process has two main steps: *argument generation* and *preferred extension generation*. First, the agents work in an independent manner, applying the specific matching approach and generating the alignment set. An alignment consists of a set of all possible correspondences between entities of two ontologies. A mapping m is described as a 5-tuple $m = (e,e',h,R,s)$, where e corresponds to an entity in the ontology 1, e' corresponds to an entity in the ontology 2, h is one of {-,+} depending on whether the mapping does or does not hold, R is the matching relation resulting from the matching between the two entities, and s is the *strength* associated to the mapping. Each mapping m is encapsulated into an argument. An *argument* $\in AR$ is a 2-tuple $x = (m,a)$, where m is a mapping; $a \in V$ is the value of the argument, depending of the agent generating that argument (i.e, syntactic, semantic or structural).

The *strength* of an argument is defined by the agent when applying the specific matching approach. After generating their set of arguments, the agents exchange with each other their arguments – the dialogue between agents consists of the exchange of individual arguments. When all agents have received the set of arguments of the each other, they generate their *attacks* set. An *attack* (or counter-argument) will arise when we have arguments for the mapping between the same entities, but with conflicting values of h. For instance, an argument $x = (m_1,L)$, where $m_1 = (e,e',+,1.0,equivalence)$, have as an *attack* an argument $y = (m_2,E)$, where $m_2 = (e,e',-,1.0,equivalence)$. m_1 and m_2 refer to the same entities e and e' in the ontologies. The argument y also represents an *attack* to the argument x.

When the sets of arguments and attacks have been produced, the agents need to define which of them must be accepted, with respect to each audience. To do this, the agents compute their preferred extensions, according to the audiences and strength of the arguments. Based on the generated preferred extensions, the sets *global* and *local* consensus are defined.

Finally, it must be point out that when two matchers indicate different matching relations for the corresponding entities, the relation indicated by the matcher with greater preference in the preference order is selected.

Table 5.2: Algorithm for defining a preferred extension (Bench-Capon and Dunne [2002]).

EXTEND(AF,attacks)
1: S:= {s ∈ AR: (∀y)(not defeats(y,s))}
2: R:= {r ∈ AR: ∃s ∈ S for which defeats(s,r)}
3: if S = ∅ then return S and Halt
4: AR':= AR \ (S ∪ R)
5: attacks':= attacks \ ((S × R) ∪ (R × AF) ∪ (AF × R))
6: Return S ∪ EXTEND(AR',attacks')

5.3.4 Argumentation algorithm

The complexity related to the definition of the preferred extensions in the S-VAF is derived of the complexity of the VAF. Such complexity arises from the possibility of cycles in the graph and the plurality of preferred extensions derives from such assumption. In the literature of AFs, the cycles can be classified according to the number of values V into *monochromatic*, if they contain arguments relating to a single value, *dichromatic* if they contain arguments relating to exactly two values, and *polychromatic* if they have more than two values. In the S-VAF, the cycles are always *polychromatic* because each argument is distinguished by a different value $v \in V$. In a VAF, as specified in (Bench-Capon [2002],Bench-Capon and Dunne [2002]), a *polychromatic* cycle has a unique, non-empty preferred extension. Such assumption is valid to a S-VAF. A unique non-empty preferred extension can be constructed by the algorithm describe in Table 5.2, relative to a value ordering (audience).

At least one argument is removed in each pass, thus, the algorithm will eventually halt. The set returned is the preferred extension. The arguments in S must be included in the preferred extension because they are no defeated. Either they were initially not defeated, or their attackers are removed in an earlier pass before they were included in S. Similarly, no argument from R can be in the preferred extension, because their inclusion in R means that they are defeated by an argument in S. The new system (AR',attacks') now contains a subset S' of arguments with no attackers in AR'. These are those arguments which were originally attacked by arguments in R, and we know that a defence to these attacks is provide by S. These arguments may therefore be included in the preferred extension.

The algorithm described in Table 5.2 is used for each matcher to compute its preferred extension. Table 5.3 describes the algorithm used to compute the *global* and *local* consensus.

5.4 Summary

In this chapter, the whole proposed agent-based model was presented. Firstly, the matcher agents were detailed. These matchers represent the broad categories of matching approaches. A differential characteristic is attributed to the proposed matchers, when compared with

Table 5.3: Algorithm for defining *global* and *local* consensus via argumentation.

Require: matchers Ag, mappings Map, audiences preference $Pref_a$
Output: *global* consensus G and *local* consensus L

1: for each Ag_i:
2: $Args_i = generateArgs(Map_i)$
3: $sendArgs(Args_i, Ag_k)$ where $i \neq k$
end for
4: for each $Args_i$:
5: if $(e_{i,j} = e_{k,j})$ and $(e'_{i,j} = e'_{k,j})$ and $(h_{i,j} = h_{k,j})$ then
6: $attacks_i(Arg_i, Arg_k)$
end for
7: $S\text{-}VAF_i = createAF(args_i, attacks_i, Pref_i)$
8: $preferred_i = EXTEND(S\text{-}VAF_i, attacks_i)$
9: $G = \{arg \in preferred_i \mid arg \in \forall\ preferred_n$ or $m_i \in arg \mid m \in \forall\ preferred_n\}$
10: $L = \{arg \in preferred_i \mid arg \in \exists\ preferred_n$ or $m_i \in arg \mid m \in \exists\ preferred_n\}$

the state-of-the-art techniques: they are able to output four kind of relation (*exactMatch*, *broadMatch*, *narrowMatch*, and *relatedMatch*), together a continuous confidence measure. Most of the state-of-the-art matcher systems output single relations, generally equivalence relations or numeric measures. Moreover, the relations are computed based on the composite labeled entities.

Next, the cooperative and argumentation-based approaches were detailed. In both approaches, the dialogue between the agent is based on the exchange of matching views. In the cooperative model, a preference voting mechanism is used to obtain the consensus between the different matchers.

On the other hand, the argumentation model evaluate arguments in favor and against mappings based on the notions of attacks and acceptability of arguments. In order to represent arguments with confidence degrees, the Strength Value Argumentation Framework (S-VAF) is proposed as an extension of the Value-based Argumentation Framework (VAF) proposed by Capon. To each argument is associated a *confidence degree*, representing how confident an agent is in the similarity of two ontology entities. Based on their preferences and confidence of the arguments, the agents compute the preferred matching sets. The arguments in such preferred sets are viewed as the set of acceptable arguments.

In the next chapter, the evaluation of the proposed models is presented.

Chapter 6

Experiments

This chapter presents the experiments to match two ontologies using the negotiation and argumentation models, which were detailed in the previous chapter. Two data sets are used: (i) a benchmark of ontologies in the bibliographic domain provided by the Ontology Alignment Evaluation Initiative (OAEI)[1]; and (ii) a real-world library case, from National Library of the Netherlands. For the OAEI benchmark, two experiment settings are specified, depending on the input used to the negotiation and argumentation models: (i) results from the proposed agents; and (ii) results from the systems that had participated in the OAEI 2007 benchmark.

The chapter is organized as follows. Section 6.1 presents the experiments using the OAEI benchmark, where the OAEI dataset is described and the experiments for the two OAEI settings are presented. Section 6.2 presents the experiments using the library case, where the dataset is described, the configuration of the matchers is presented, and the experiments are detailed.

6.1 Benchmark Case

6.1.1 Dataset description

The Ontology Alignment Evaluation Initiative (OAEI) is a coordinated international initiative to establish a consensus for evaluation of the methods for ontology matching. It organizes evaluation campaigns that aims to evaluate ontology matching technologies, through the controlled experimental evaluation of the techniques performances. Specifically, the goals of the OAEI are: (i) assessing strength and weakness of matching systems; (ii) comparing performance of techniques; (iii) increasing communication among algorithm developers;

[1] http://oaei.ontologymatching.org/

and (iv) improving evaluation techniques.

A systematic benchmark[2] is provided by the OAEI community. The goal of this benchmark is to identify the areas in which each algorithm is strong and weak. The test is based on one particular (reference) ontology dedicated to the domain of bibliography and a number of alternative ontologies of the same domain for which alignments are provided. The reference ontology contains 33 named classes, 24 object properties, 40 data properties, 56 named individuals and 20 anonymous individuals.

Basically, the reference ontology is matched with different alterations of itself. There are six categories of alterations:

1. *name* (label), where the name of entities can be replaced by random strings (R), synonyms (S), name with different conventions (N), or names are described using another language than English (F);
2. *comments*, which can be suppressed (N) or translated in another language (F);
3. *specialization hierarchy*, where the hierarchy can be suppressed (N), expanded (E) or flattened (F);
4. *instances*, which can be suppressed (N);
5. *properties*, which can be suppressed (N) or having the restrictions on classes discarded (R);
6. *classes*, which can be expanded (E) – replaced by several classes – or flattened (F).

The data sets involving such alterations are grouped into three categories: (a) *concept test* (tests 101, 102, 103 and 104); (b) *systematic* (tests 201 – 266); and (c) *real ontologies* (tests 301, 302, 303, and 304). *Systematic* tests involve alterations in the proprieties, classes, hierarchy, and instances of the alternative ontologies, as the categories of alterations commented above. Table 6.1 characterizes the tests according to the kind of alterations on the alternative ontologies.

6.1.2 Evaluation measures

As a mapping quality evaluation, the measures of precision, recall and f–measure are used. Such measures are derived from a contingency table (Table 6.2).

Precision (P) is defined by the number of correct automated mappings (m_{++}) divided by the number of mappings that the system had returned ($m_{++} + m_{+-}$). It measures the system's

[2]http://oaei.ontologymatching.org/2007/benchmarks/

Table 6.1: OAEI Benchmark Dataset

Test	Label	Comments	Hierarchy	Instance	Properties	Class	Description
101							Reference alignment
102							Irrelevant ontology
103							Language generalization
104							Language restriction
201	R						No names
202	R	N					No names, no comments
203		N					No comments
204	C						Naming conventions
205	S						Synonymous
206	F	F					Translation
207	F						Translation
208	C	N					
209	S	N					
210	F	N					
221			N				No specialization
222			F				Flattened hierarchy
223			E				Expanded hierarchy
224				N			No instance
225					R		No restrictions
228					N		No properties
230						F	Flattened classes
231						E	expanded classes
232			N	N			
233			N		N		
236				N	N		
237			F	N			
238			E	N			
239			F		N		
240			E		N		
241			N	N	N		
246			F	N	N		
247			E	N	N		
248	N	N	N				
249	N	N		N			
250	N	N			N		
251	N	N	F				
252	N	N	E				
253	N	N	N	N			
254	N	N	N		N		
257	N	N		N	N		
258	N	N	F	N			
259	N	N	E	N			
260	N	N	F		N		
261	N	N	E		N		
262	N	N	N	N	N		
265	N	N	F	N	N		
266	N	N	E	N	N		
301							BibTex MIT[3]
302							BibTex UMBC[4]
303							BibTex Karlsruhe[5]
304							INRIA[6]

Table 6.2: Contingency table for binary classification.

	manual h = +	manual h = -
output h = +	m_{++}	m_{+-}
output h = -	m_{-+}	m_{--}

correctness or accuracy. *Recall* (R) indicates the number of correct mappings returned by the system divided by the number of manual mappings ($m_{++} + m_{-+}$). It measures how complete or comprehensive the system is in its extraction of relevant mappings. *F–measure* (F) is a weighted harmonic mean of precision and recall.

$$P = \frac{m_{++}}{(m_{++} + m_{+-})}, \quad R = \frac{m_{++}}{(m_{++} + m_{-+})}, \quad F = \frac{(2 * P * R)}{(P + R)}$$

To measure the global performance of the system, the measures of *macro-averaging* and *micro-averaging* (Joachims [2002]) are used. Such measures are often useful to compute the average performance of a system over multiple test sets, where the results of n binary tasks can be averaged to get a single performance value. *Macro-averaging* corresponds to the standard way of computing an (arithmetic) average. The performance (i.e. precision and recall) is computed separately for each of the n test. The average is computed as the arithmetic mean of the performance measure over all tests. *Micro-averaging* averages the contingency tables of the various tests. For each cell of the table, the arithmetic mean is computed – m_{++}^{avg}, m_{+-}^{avg}, m_{-+}^{avg}, m_{--}^{avg} – and the performance is computed from this averaged contingency table. For the precision, *macro-averaging* and *micro-averaging* imply:

$$P^{macro} = \frac{1}{n}\sum_{i=1}^{n} P_i, \quad P^{micro} = \frac{m_{++}^{avg}}{(m_{++}^{avg} + m_{+-}^{avg})}$$

Macro-averaging gives equal weight to each test whereas micro-averaging gives equal weight to each mapping (example).

For all comparative results, a significance test is applied, considering a confidence degree of 95%. Such comparison is indicated in bold face in the tables below. When there is reference for results *slightly* better, we mean that some true positive mappings are retrieved while some false positive mappings are discarded, however without having so significantly differences in the results.

6.1.3 Benchmark Setting 1: Using the Proposed Matchers

6.1.3.1 Matchers configuration

In the first experiment setting, the proposed matchers (Section 5.1) are used. Four kinds of matcher agents are considered: (i) *syntactic* matcher (Section 5.1.1), (ii) *semantic* matcher (Section 5.1.2), (iii) *structural* matcher (Section 5.1.3), and (iv) *attribute* datatype matcher. The *syntactic* matcher applies a threshold of 0.8 when computing the edit distance (i.e., strings with edit distance below this value do not match). The *semantic matcher* is based on the Java WordNet Interface API[7], which is an interface to the WordNet database. WordNet database version 2.1 is used.

The *structural matcher* is implemented using the syntactic and semantic similarity measures, i.e., two entities match if they are syntactically similar (using edit distance) or if they are semantically related (synonymous), respectively. Two values are used to the parameter n (number of considered super-classes), 1 and 2. Thus, four structural matchers are specified: (i) structural-syntactic with $n = 1$ (SSyn1); (ii) structural-syntactic with $n = 2$ (SSyn2); (iii) structural-semantic with $n = 1$ (SSem1); and (iv) structural-semantic with $n = 2$ (SSem2).

The *attribute* datatype matcher is a simple matcher based on the comparison of the datatype used to specify the properties. Two entities (properties) match if they have the same datatype. For instance, "title" and "name" are two properties that match because both have the type *string*.

A pre-processing step that involves tokenization and lemmatization processes was applied. For each token, its lemma is obtained using the Tree-Tagger-English tool[8].

The matchers are implemented in Java for Linux, version 1.5.0, and the matching process run on a Intel(R) Core (TM) Duo CPU 2GHz, 2038MB. In average, each matchers takes 40 minutes to generate its mappings (the syntactic matcher consume less time), while negotiation and argumentation take 40 seconds and 2 minutes, respectively.

6.1.3.2 Individual matchers

This section presents the individual results of the proposed matchers. The results of *syntactic* and *semantic* matchers consider both entities that are classes and attributes, while the *structural* matcher is applied to the classes, and the *attribute* matcher refers to the entities that are attributes. The results for the structural matcher refer to the better results obtained from one of the four settings (SSyn1, SSyn2, SSem1, SSem2). For all tests, the SSyn1 has similar performance than the SSem1, and better results than the other structural configurations. This

[7] http://www.mit.edu/ markaf/projects/wordnet/

[8] http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/DecisionTreeTagger.html

Table 6.3: Individual matcher results.

	Syntactic			Semantic			Structural	Attribute
	C	A	All	C	A	All	C	A
P^{macro}	**1.0**	**0.80**	**0.90**	**0.99**	**0.71**	**0.84**	**1.0**	0.02
P^{micro}	**1.0**	**0.99**	**0.99**	**1.0**	0.88	0.93	**1.0**	0.03
R^{macro}	**0.53**	0.54	0.53	**0.51**	0.48	0.49	0.34	**0.95**
R^{micro}	**0.53**	0.55	**0.54**	**0.50**	0.49	**0.50**	0.35	**0.97**
F^{macro}	**0.85**	**0.59**	0.67	**0.82**	**0.52**	0.62	0.66	0.05
F^{micro}	**0.69**	**0.70**	0.70	**0.67**	**0.63**	0.65	**0.52**	0.06

fact arrives because the OAEI ontologies are structurally simple. Thus, the results below refer to the SSyn1 configuration.

Table 6.3 shows the results for each matcher, where the columns (C), (A), (All) refer to the values for entities that are classes, attributes, and both of them, respectively. For the sake of brevity, the values of macro-averaging and micro-averaging are presented. A detailed corresponding table can be found in the Appendix B (Table B.1). Considering that the results of the benchmark consider only the relation of "exactMatch", only the mappings with such relation are take into account.

As shown in Table 6.3, when considering the classes (C), the syntactic and semantic matchers have similar results of F–measure, while the F–measure^{macro} of syntactic matcher is significatively better than the results returned by the structural matcher. This is due the fact that in the majority of the tests (except 248-266), the reference ontology is matched with alterations of itself with few syntactic variations on the labels of the entities. When analyzing the attributes (A), the syntactic matcher presents again the better results for F-measure, and the attribute matcher shows that it is insufficient to the problem: several attributes have the same datatype but they refer to different entities. For all entities (All), the results refers to the same analysis for both classes and attributes.

Examination of the results in the different test groups (Table B.1 in Appendix B) shows that all matchers produce comparable results. The exceptions (for worst results) are the *systematic* tests 201, 202, 248, 249, 250, 251, 252, 253, 254, 257, 258, 259, 260, 261, 262, 265, and 266. For the tests 201 and 202, the concept labels of the ontologies have been replaced by random strings and the matchers are based on the analysis of the labels, thus performing not well. For the tests 248-266 (shown in Appendix B), different name conventions for the labels are used and the ontology hierarchy is flattened, leading to necessity of matching techniques that use alternative input elements than labels and hierarchy (i.e., matching techniques different than the used by the proposed matchers). As stated in (Hu et al. [2006]), these tests are the most challenging ones, and it was extremely difficult to recognize the correct alignments. For *complex* tests, structural and attribute matchers produces the worst F–measure.

6.1.3.3 Baseline, negotiation and argumentation results

The use of the negotiation and argumentation models aims to obtain a consensus between the matchers, improving or balancing the individual results. This section presents the results using such models, considering as input the results from the proposed matchers. First, different values of strength for the arguments that represent a negative mapping (h = -) are considered to verify the behavior of the argumentation model. Next, the negotiation and the argumentation models are analyzed against the baseline – which is composed by the union of all individual mappings – and individual results of each matcher.

In the negotiation model, when positive and negative mappings receive the same number of votes, the vote of *syntactic* matcher is used to solve the conflict.

The argumentation result contains only the arguments objectively acceptable, and the audiences represent the following complete preference order, which has been defined according to the individual performance of the matchers (Table 6.3): *syntactic* audience – syntactic > semantic > structural; ***semantic*** audience – semantic > syntactic > structural; ***structural*** audience – structural > syntactic > semantic; and ***attribute*** audience – attribute > syntactic > semantic.

Firstly, two values are used to represent the strength of counter-arguments of a positive mapping, 0.5 and 1.0. Table 6.4 shows the results. The detailed corresponding table can be found in Appendix B (Table B.2).

Considering that the matchers output arguments for positive mappings with strength between 0.80 and 1.0, using 0.5 as strength for a negative mapping does not represent attacks for the positive mappings. When the matchers have good performance, this results better values of recall (all positive mappings are selected). It is the case of the classes (C) in Table 6.4, where significant difference is observed between the 0.5 and 1.0 F–measures^{macro}. On the other hand, if there are matchers with poor performance, the negative mappings from the matcher with good performance do not represent attacks to the false positive mappings, resulting in a low precision. It is the case of the attributes (A), where the true negative mappings from syntactic and semantic matchers do not attack the false positive mappings from the weak attribute matcher (with significant differences between both F–measure^{macro} and F–measure^{micro}).

When using a value of 1.0 for the negative mappings, the false positive mappings from the bad matchers – case attributes (A) – are possibly attacked or not selected as objectively acceptable (the false positive mapping is not acceptable for the audience of the true positive mapping). In this way, the precision is significantly better. On the other hand, the resulting recall represents the lower recall of the good matchers – case classes (C). It is the case of the structural matcher (C), which possibly have arguments that successfully attack the arguments from the syntactic and semantic matchers or the arguments are not objectively

Table 6.4: Different values of strength for the negative mappings – proposed matchers.

	0.5			1		
	C	A	All	C	A	All
P^{macro}	**1.0**	0.02	0.15	**1.0**	**0.80**	**0.90**
P^{micro}	**1.0**	0.03	0.04	**1.0**	**0.99**	**0.99**
R^{macro}	**0.54**	**0.92**	**0.70**	0.33	0.49	0.39
R^{micro}	**0.53**	**0.95**	**0.77**	0.33	0.49	0.42
F^{macro}	**0.86**	0.05	0.21	0.63	**0.55**	**0.54**
F^{micro}	**0.69**	0.05	0.08	0.50	**0.66**	**0.59**

acceptable. For (All) setting, both F–measure^{macro} and F–measure^{micro} using 1.0 for the negative mappings, are better than the F–measures when using 0.5.

A normalized value of strength for the positive mappings was verified (i.e., all the positive mappings with strength 1.0). Considering that the matcher already output values in the range [0.8, 1.0], the results were the same of those shown in Table 6.4.

Next, the results regarding strength 1.0 are considered. Table 6.5 shows the results of the baseline, negotiation, and argumentation. The detailed corresponding table can be found in Appendix B (Table B.3). As expected, the baseline has higher recall than the output of the argumentation and negotiation models. For classes (C), there is significant difference between the recall of the baseline and the recall of the argumentation (specially R^{macro}), while for the attributes (A), there is significant difference between the recall of the baseline and both recalls of negotiation and argumentation models. On the other hand, the baseline produces low values of precision, specially in the presence of several false positive mappings – case of using the attribute matcher, columns (A) and (All). In these cases, precision of the negotiation and argumentation is significantly better than the precision of the baseline (i.e., using negotiation or argumentation, the false positive mappings can be discarded). For the more interesting setting, (All), this produces better values of F–measure for the negotiation and argumentation models than the baseline.

When comparing the negotiation and argumentation models, there is no significant difference between the values of precision and recall, and consequently F–measure (the exception is the F–measure^{macro} for classes (C), where negotiation performs better). However, a false positive mapping that is not acceptable for one matcher can be acceptable by voting in the negotiation process, what can not happen in the argumentation process (the corresponding counter-argument is not objectively acceptable). Thus, the argumentation process can filter some false positive mappings (improving the precision) but, on the other hand, can eliminate true positive mappings which are acceptable only for one matcher, reducing the recall.

Looking for each group of tests, the best performance is achieved for *concept* tests (high performance of the matchers). For *systematic* tests, negotiation improves slightly the recall while argumentation improves the precision (specially for the more interesting case – (All)

Table 6.5: Baseline, negotiation, and argumentation (strength 1.0) results – proposed matchers.

	Baseline			Negotiation			Argumentation		
	C	A	All	C	A	All	C	A	All
P^{macro}	**0.99**	0.02	0.15	**1.0**	**0.71**	**0.84**	**1.0**	**0.80**	**0.90**
P^{micro}	**1.0**	0.03	0.04	**1.0**	**0.89**	**0.93**	**1.0**	**0.99**	**0.99**
R^{macro}	**0.54**	**0.92**	**0.70**	**0.52**	0.54	0.52	0.33	0.49	0.39
R^{micro}	**0.53**	**0.95**	**0.77**	**0.52**	0.55	0.54	0.33	0.49	0.42
F^{macro}	**0.86**	0.05	0.21	**0.84**	**0.56**	**0.65**	0.63	**0.55**	**0.54**
F^{micro}	**0.69**	0.05	0.08	**0.68**	**0.68**	**0.68**	**0.50**	**0.66**	**0.59**

setting). Such behavior is also achieved in the all cases of *concept* tests. For tests 248-266 (shown in Appendix B, Table B.3), considering that in such tests both the lexical information and the structure of the target ontologies have been heavily changed, the matchers perform not well resulting in a low performance of both negotiation and argumentation models. For *real* cases, the previous analysis is corroborated.

Analyzing the results of negotiation and argumentation with the individual results, when considering only the classes – column (C) in Tables 6.5 and 6.3, respectively – the matchers produce similar sets of positive mappings (i.e., the baseline produces similar precision and recall than the individual matchers), and the output of the negotiation and argumentation processes is similar to the individuals ones. For the attributes (A), using negotiation and argumentation produce a considerable improvement in the results when looking for the attribute matcher (A). This matcher produces a great number of false positive mappings, which are not produced by other matchers (i.e., so different properties are described by the same datatype). Thus, using negotiation and argumentation part of these false positive mappings are filtered out.

6.1.3.4 Comparison with Laera et al. argumentation model

The results of negotiation and argumentation (Table 6.5) are compared with the results of the state-of-the-art Laera's argumentation model (Table 6.6), which are available in (Laera et al. [2007]). In the model proposed by Laera, two agents argue on the mappings provided by an Ontology Mapping Repository (OMR). The agents' preferences ($Pref_1$ and $Pref_2$) are chosen on the basis of the ontological information. In the majority of the available test cases, the terminological value is preferred on both $Pref_1$ and $Pref_2$.

The results in the different groups shows that for the *conceptual tests* (101-104), negotiation and argumentation produce significant better results than the Laera's model. This is directly related to the quality of the mappings provided by the proposed matchers in opposition to the mappings provided by the OMR.

In the *systematic tests*, negotiation performs better in the majority of the cases. For tests 201

Table 6.6: Comparison with Laera et. al. argumentation model.

	Negotiation	Argumentation	Laera Argumentation
Test	F–measure	F–measure	F–measure
101	**0.97**	**0.97**	0.83
103	**0.97**	**0.96**	0.83
104	**0.97**	**0.96**	0.83
201	**0.02**	**0.02**	**0.02**
202	**0.02**	**0.02**	**0.02**
204	**0.77**	0.73	**0.88**
205	**0.28**	**0.23**	0.0
206	**0.46**	0.27	0.0
221	**0.96**	0.81	**0.89**
222	**0.97**	0.88	**0.91**
223	**0.96**	**0.90**	0.64
224	**0.97**	**0.97**	**0.91**
225	**0.97**	**0.97**	**0.91**
228	**1.0**	**0.98**	0.80
230	**0.96**	0.96	0.71
301	0.35	0.03	**0.68**
302	**0.57**	0.34	**0.62**
303	**0.59**	0.33	**0.57**
304	**0.75**	0.59	**0.78**
F^{macro}	**0.71**	**0.63**	**0.62**

and 202, all approaches performs similarly less well due to a lack of true positive mappings provided by the matchers and OMR (in these test ontologies, the concept labels have been replaced by random strings). In the Laera's model, for tests 205 and 206 the argumentation produces a 0 F-measure because the information in the two ontologies causes the agents to select directly opposing preferences, which leads to an inability to reach agreement on many of the mappings. This situation is avoided in the proposed negotiation and argumentation models, by voting and attacking relations with strength, respectively. In the *complex tests*, the Laera's model performs better than negotiation and argumentation in test 301, but similar results are obtained for the other tests, specially for the negotiation model.

In all, negotiation performs better than the argumentation models. However, the quality of the results is related with the quality of the mappings. A more fair comparison could be to use the same set of initial mappings as input for both three models and then to compare the final results.

6.1.3.5 Comparison with OAEI matchers

When comparing the negotiation and argumentation models (Table 6.5) with the results of the OAEI 2007 matchers (Table 6.7), there is significant better result only for negotiation than TaxoMap system, although the argumentation and negotiation models improve slightly the results of OntoDNA and TaxoMap systems.

Examination of the results in the different test groups (Tables B.3, B.4 and B.5) shows that for *conceptual* tests the values of F–measure of negotiation and argumentation (column All in Table B.3) are in a similar range to the OAEI systems (specially recall of negotiation and precision of argumentation). For *systematic* tests, negotiation and argumentation produce results below the OAEI systems, specially in the cases 201, 202, 248-266, due the limitation of the proposed matchers to use hierarchy and label-based techniques. Such tests are responsible for the low F–measure^{macro} and F–measure^{micro}. Finally, for *complex* tests, the precision produced by the negotiation and argumentation models is higher than the corresponding values produced by Ola, RiMON, and Sema, while the recall is lower.

6.1.4 Benchmark Setting 2: OAEI Matchers

6.1.4.1 Matchers configuration

The group of OAEI matchers are the participants of the OAEI Benchmark Track 2007[9]: ASMOV, DSSim, Falcon, Lily, Ola, OntoDNA, PriorPlus, RiMON, Sambo, SEMA, TaxoMap, and XSOM. These systems are described in Section 4.1.

6.1.4.2 Individual matchers results

Table 6.7 shows the results for each OAEI matcher. The detailed corresponding tables can be found in Appendix B (Tables B.4 and B.5). These results out a group of systems, ASMOV, Lily, Falcon, OLA, PriorPlus, and RiMOM which seem to perform the tests at the highest level of quality (Euzenat et al. [2007]). Of these, ASMOV, Lily and RiMOM seem to have slightly better results than the three others. A previous comparison between such systems is presented in Section 4.1.4.

6.1.4.3 Baseline, negotiation and argumentation results

Firstly, different values of strength for the arguments that represent a negative mapping (h = -) are considered to verify the behavior of the argumentation model. Next, the negotiation and the argumentation models are analyzed against the baseline and individual results.

In the negotiation model, when positive and negative mappings receive the same number of votes, the vote of ASMOV is used to solve the conflict.

The argumentation result contains only the arguments objectively acceptable, and the audiences represent the following complete preference order (template), which has been defined

[9]in http://oaei.ontologymatching.org/2007/results/

Table 6.7: Individual matcher results - OAEI systems

	ASMOV	DSSim	Falcon	Lily	Ola	OntoDNA
P^{macro}	0.93	0.97	0.93	0.94	0.88	0.54
P^{micro}	0.95	0.98	0.92	0.96	0.89	0.83
R^{macro}	0.84	0.64	0.81	0.85	0.81	0.42
R^{micro}	0.90	0.64	0.86	0.89	0.87	0.49
F^{macro}	0.87	0.71	0.84	0.88	0.83	0.54
F^{micro}	0.92	0.77	0.89	0.92	0.88	0.62
	PriorPlus	RiMON	Sambo	SEMA	TaxoMap	XSom
P^{macro}	0.89	0.95	0.89	0.89	0.93	0.72
P^{micro}	0.93	0.95	0.98	0.90	0.92	0.76
R^{macro}	0.79	0.83	0.55	0.72	0.27	0.66
R^{micro}	0.81	0.86	0.56	0.74	0.21	0.70
F^{macro}	0.82	0.86	0.69	0.76	0.58	0.73
F^{micro}	0.86	0.91	0.71	0.81	0.34	0.73

according to the individual performance of the matchers: *ASMOV* audience – ASMOV > Lily > RiMON > Falcon > Ola > PriorPlus > Sema > DSSim > XSom > Sambo > OntoDNA; *Lily* audience – Lily > ASMOV > RiMON > Falcon > Ola > PriorPlus > Sema > DSSim > XSom > Sambo > OntoDNA; and so on.

Two values are used to represent the strength of the counter-argument of a positive mapping, 0.5 and 1.0. Table 6.8 shows the results. The detailed corresponding table can be found in Appendix B (Table B.6).

Basically, the OAEI matchers produce arguments for positive mappings with strength between 0.80 and 1.0, using 0.5 as strength for a negative mapping does not represent attacks for the positive mappings. As the matchers have good performance (as shown in Tables B.4 and B.5, Appendix B), this results better values of recall (the majority of the true positive mappings are selected). However, some false negative mappings are selected because true negative mappings do not represent attacks, resulting lower precision.

When using a value of 1.0, the false positive mappings from the matchers with lower performance are possibly attacked or not selected as objectively acceptable (the false positive mapping is not acceptable for the audience of the true positive mapping). In this way, the precision is better. On the other hand, the resulting recall represents the lower recall of the matchers. Moreover, a notable problem when using the value 1.0 in the absence of mappings for one matcher is that if all others have true positive mappings with strength below 1.0, such true positive mappings are successfully attacked by the negative mappings. It is what occurs with the TaxoMap matcher, for the tests 248-266 (Table B.6, Appendix B).

Next, the results of baseline, negotiation, and argumentation are compared (Table 6.9). The detailed corresponding table can be found in Appendix B (Table B.7).

When compared with the baseline, the negotiation and argumentation models eliminate false

Table 6.8: Different values of strength for the negative mappings and *original* strength for positive mappings – OAEI matchers.

	0.5	1.0
P^{macro}	0.70	**1.0**
P^{micro}	0.67	**1.0**
R^{macro}	**0.89**	0.13
R^{micro}	**0.93**	0.13
F^{macro}	**0.77**	0.40
F^{micro}	**0.78**	0.22

positive mappings, resulting slightly better values of precision than the baseline. However, significant results are obtained using negotiation. For recall, there are no significant differences for the results for both three settings. In average, looking for the F–measures values, the results are significantly better for the negotiation model.

Comparing negotiation and argumentation, taking into account that 0.5 is used as strength for the negative mappings, as expected the precision can be lower, while recall is higher. Thus, negotiation achieves higher precision than argumentation, while argumentation produces better values of recall.

An observed behavior, specially in *systematic* tests 248-266, is that the majority of the matchers returns a low number of true positive mappings and sometimes the sets of produced mappings are disjunctive. By voting, only the true positive mappings returned by the majority of the matchers are selected. This implies a high precision, but a lower recall, due the low number of mappings produced by the matchers. On the other hand, when using argumentation the attacking relations can overlap small sets of mappings (for example, ASMOV has an argument with strength 1 for a true positive mapping that successfully attacks an argument with strength 0.5 from TaxoMap representing a false negative mapping). Thus, the recall of argumentation is higher while precision is lower (the false positive mappings are not attacked, because the strength of the false negative mappings is possibly lower – 0.5 – than the strength of the false positive mappings). For *real* tests, negotiation produces significant better values of precision than the argumentation model, and similar values of recall, resulting better values of F–measure.

Table 6.9: Baseline, negotiation, and argumentation (strength 0.5) – OAEI matchers.

	Baseline	Negotiation	Argumentation
P^{macro}	0.61	**0.99**	0.70
P^{micro}	0.57	**0.99**	0.67
R^{macro}	**0.90**	0.79	**0.89**
R^{micro}	**0.94**	0.84	**0.93**
F^{macro}	0.71	**0.88**	0.77
F^{micro}	0.71	**0.91**	0.78

When analyzing the results of negotiation and argumentation (Table 6.9) with the individual

results (Table 6.7), using negotiation slightly improves the precision of all OAEI matchers, while having significant similar results than the best OAEI matchers (ASMOV, Lily, RiMON, and OLA). With argumentation, a similar behavior is found for recall, where it slightly improves the precision of all OAEI matchers, but has significant similar results than the best OAEI matchers (ASMOV, Lily, RiMON, and OLA). In terms of F–measure, negotiation produces results in the range of the those achieved by the best OAEI matchers. However, if we consider real ontologies only, negotiation outperforms slightly the best matcher.

6.2 Real-World Library Case

The second set of experiments involves matching two real thesaurus provided by National Library of the Netherlands. This dataset corresponds to the Library track of the OAEI 2007 campaign[10]. The evaluation is an application dependent one due the fact that there is no reference alignment available.

6.2.1 Dataset description

The National Library of the Netherlands maintains two large collections of books: the Deposit Collection, containing all the Dutch printed publications (one million items), and the Scientific Collection (1.4 million books). Each collection is annotated – *indexed* – using its own controlled vocabulary. The Scientific Collection is described using the GTT thesaurus, a huge vocabulary containing 35.194 general concepts. The books in the Deposit Collection are describes against the Brinkman thesaurus, which contains a large set of headings (5.221) for describing the overall subjects of books. Among the 2.4 Million books in the two collections, 250K are actually dually annotated by both thesauri. Concepts are provided with preferred labels, synonymous, extra hidden labels, and structural information, in the form of narrower, broader, and related links.

6.2.2 Evaluation measures

The evaluation is done in an annotation translation scenario supporting the re-indexing of GTT-indexed books with Brinkman concepts (see Isaac et al. [2008a] for more details). This is useful if GTT is dropped: a huge volume of legacy data has to be converted to the remaining annotation system. The evaluation is based on a tool that interprets the correspondences provides by the different matchers so as translate existing GTT annotations into equivalent Brinkman annotations.

[10]http://oaei.ontologymatching.org/2007/library/

The simple concept-to-concept correspondences of the matchers are transformed into complex mapping rules that associate one GTT concept and a set od Brinkman concepts. The set of GTT concepts attached to each book is then use to decide whether these rules are *fired* for this book. If the GTT concept of one rule is contained by the GTT annotation of a book, then the rule is fired. As several rules can be fired for a same book, the union of the consequents of these rules forms the translated Brinkman annotation of the book.

To carry out the evaluation, it is used the 250K dually annotated books (golden standard). It is measured (i) how many translated concepts are correct over the annotation produced for the books on which rules are fired (P_a), (ii) how many correct Brinkman annotation concepts are found for all books in the evaluation set (R_a, and (iii) the combination of these two, namely a Jaccard overlap measure between the produced annotation an the correct one (J_a).

$$P_a = \frac{\sum \frac{\#correct}{|B_t|}}{\#books\_fired}, \quad R_a = \frac{\sum \frac{\#correct}{|B_o|}}{\#all\_books}, \quad J_a = \frac{\sum \frac{\#correct}{|B_o \cup B_t|}}{\#all\_books}$$

where $\#correct$ is the number of the translated Brinkman concepts which are actually used, B_o and B_t are the original and transtlated Brinkman annotation, respectively.

6.2.3 Matchers configuration

Two group of matchers are used to carry out the experiments. The first group involves three OAEI matchers that had participated of the OAEI Library Track 2007: Falcon, DSSim, and Silas. These tools are *hybrid*, as they use several alignment techniques in an integrated process. These systems are described in the Section 4.1. It is important to point out they mainly return equivalence (*exactMatch*) mappings, except Silas, which provides a significant number of *related* mappings.

The second group is composed by alternative matchers, namely the syntactic matcher described in Section 5.1.1, the SKOS Dutch lexical, and the co-occurrence matcher.

As many lexical mappers are only dedicated to English, the STITCH and CHOICE Dutch research projects have developed the SKOS matcher, a Dutch-specific lexical matcher. It is based on the CELEX morphology database, which allows to recognize lexicographic variants and morphological components of a word form. This mapper produces *exact* matches between concepts, but also hierarchical *broader* matches, based on the morphological (*resp.* syntactic) decomposition of the words (*resp.* expressions) that label them. The different lexical comparison methods used by this mapper give rise to different confidence measures: using exact string equivalence is more reliable than using lemma equivalence. Also, the mapper considers the status of the lexical features it compares. The concepts that are aligned are described according to the SKOS model (Miles and Bechhofer [2008]), which means that their labels can be either *preferred* or *alternative*. The latter ones can be near synonyms.

For two concepts, any comparison based on them is therefore considered less reliable than a comparison based on alternative labels. The combination of these two factors — different comparison techniques and different features compared — results in a grading of the produced mappings, which can be used as a confidence level.

The co-occurrence based matcher is an instance-based mapper has been developed in the context of the STITCH project Isaac et al. [2008b]. In the application context, the instances of a concept c, noted $e(c)$, are the set of books related to this concept via a `subject` annotation property. For each pair of concepts, the overlap of their instance sets is measured and considered as the confidence level for an equivalence *exactMatch* relation. The measure is adapted from the standard Jaccard similarity, so that it assigns a smaller score to less frequently (co-)occurring concepts:

$$overlap_i(c_1, c_2) = \sqrt{|e(c_1) \cap e(c_2)| \times (|e(c_1) \cap e(c_2)| - 0.8)} \; / \; |e(c_1) \cup e(c_2)|.$$

The 0.8 factor has been chosen so that evidence for concepts that have just one co-occurring instance is weighed as much as mapping two concepts would get when a large number of concepts have 20% in their intersection. This choice is relatively arbitrary, but the obtained measure has shown to perform well on previous experiments carried out with the dataset considered in this paper.

6.2.4 Individual matcher results

Table 6.10 shows the results for the individual matchers. Such results refer the evaluation when using all type of mappings in order to produce the rules (i.e., *exact*, *broad*, *narrow*, and *related*). As shown in Table 6.10, Falcon, SKOS, and Silas perform similar (J-a) and much ahead of DSSim, while the syntactic matcher perform better than Co-occurrence and DSSim.

Table 6.10: Individual matcher results – Real-library case.

Matcher	P-a	R-a	J-a
DSSim	0.13	0.09	0.07
Falcon	**0.52**	0.36	**0.30**
Silas	0.45	0.42	**0.31**
SKOS	0.40	0.43	**0.29**
Co-occurrence	0.13	**0.79**	0.12
Syntactic	0.31	0.43	0.24

As commented in (Euzenat et al. [2007]), half of the translated concepts are not validated, and more than 60% of the real Brinkman annotation is not found. The correspondences from Falcon are mostly generated by lexical similarity. This indicates that lexically equivalent correspondences alone do not solve the annotation translation problem. It also confirms the necessity of the combination with the alternative matchers.

6.2.5 Baseline, negotiation and argumentation results

For the experiments using argumentation, a complete preference order is used between the audiences representing the matchers. The order is based on the individual performance of the matchers. For OAEI group of matchers, for instance, the Falcon's audience has the following order: Falcon > Silas > DSSim. For the alternative matchers, the co-occurrence's audience has the order: SKOS > syntactic > co-occurrence. For the combination of the two group de matchers, a template order is the following: Falcon > Silas > SKOS > syntactic > co-occurrence > DSSim. The strength for the negative mappings (h = -) is 1.0.

Table 6.11 shows the results of baseline, negotiation and argumentation for three combinations of matchers: OAEI matchers, alternative matchers (Skos + syntactic + co-occurrence), and all matchers. As expected, negotiation and argumentation models eliminate false positive mappings (and also true positive) when compared with the baseline results. This results higher precision and lower recall than the baseline.

Argumentation is more selective. When a counter-argument with strength 1 is generated for one matcher, it successfully attacks the positive arguments issued by the matchers with lesser preference. When each audience privileges the arguments produced by the matcher it represents, this amounts filter out from the objectively acceptable mappings all those belong the intersection of mappings with strength 1. This implies an expected great increasing in precision and a decrease in recall, compared with the baseline. For the OAEI combination (as well as for All, which includes it), the intersection is very small (caused by DSSim missing a lot of good mappings) which causes recall to be very low. For the alternative matchers, the intersection is larger, explaining an improvement for recall.

Using negotiation, it is possible to retrieve significant part of the intersection sets of all mappings, considering the selection of the mappings based on voting. For example, if both Falcon and Silas have a argument in favor a positive mapping, independently of the strength of a possible argument against the mapping from DSSim, the positive mapping is selected.

Table 6.11: Baseline, negotiation and argumentation on combined matchers – Real-library case.

	OAEI			Alternative			All		
	P-a	R-a	J-a	P-a	R-a	J-a	P-a	R-a	J-a
Baseline	0.32	**0.46**	**0.26**	0.13	**0.79**	0.12	0.12	**0.80**	0.11
Negotiation	**0.53**	0.32	**0.28**	0.45	0.44	**0.32**	**0.53**	0.38	**0.31**
Argumentation	0.52	0.07	0.07	**0.52**	0.37	**0.31**	**0.53**	0.07	0.06

For the more interesting case which involves the combination of all matchers, the precision of both negotiation and argumentation are similar. However, significant differences are found for the recall. In all, negotiation achieves the better values of J-a, which are similar to the best individual matcher, improving significantly the results of the worst matchers.

6.3 Discussion

Using 0.5 or 1.0 for the strength of negative mappings is a trade-off between precision and recall. For 0.5, good values of recall are achieved, while precision is lower. It is due the fact that the arguments with such strength do not represent attacks for the arguments representing true and false positive mappings. On the other hand, using 1.0 as strength good values of precision are obtained, while recall is lower. It happens because the false positive mappings and some true positive mappings are successfully attacked or not selected as objectively acceptable. When taking into account the quality of the individual matchers, in the presence of matchers with low performance (case attributes and all settings using the proposed matchers), it is preferred to use a strength of 1.0, while for good matchers (case OAEI matchers), 0.5 achieves good results.

As expected, baseline produces higher recall (all true positive mappings are retrieved) than argumentation and negotiation, while precision is lower (all false positive mappings are retrieved). By negotiation and argumentation, false positive mappings can be are filtered out, improving the precision, while true positive mappings are also discarded, reducing the recall. This behavior is accentuated in the presence of bad matchers (case attributes and all settings for the proposed matchers, and library case).

In average, negotiation performs better than argumentation (regarding F–measure values). For the setting using the proposed matchers, although there are no significant differences in the results, argumentation slightly improves the precision while negotiation achieves higher values of recall (expected behavior when using 1.0 as strength for a negative mapping). By voting the majority of the true positive mappings are retrieved, while by argumentation, a possible false positive mapping is not objectively accepted (i.e., only one matcher can not accept such false positive mapping). However, for the setting using the OAEI matchers, a different behavior is verified. As a value of 0.5 is used as strength for the negative mappings, as expected, the precision is lower, while recall is higher. Thus, negotiation achieves higher precision than argumentation, while argumentation produces better values of recall. For the library case, although negotiation and argumentation produce similar values of precision, argumentation is more selective in terms of recall (specially when the intersection of good mappings is small due the presence of matchers that not perform well).

It is important point out that only the arguments objectively acceptable are considered in the evaluation process. It means that only the mappings strictly acceptable for all matchers are evaluated.

Analyzing the results of the individual matchers, the consensus achieved by the cooperative models is a balancing between the individual results. By consensus not exactly there is improvement in all individual results, but a intermediary values near of the best matcher and a considerable improvement in the worse matchers. For the case of the proposed matchers,

significant improvement is achieved in the cases of attributes (A) setting. For the classes (C) setting, the matcher output mappings within a intersection set, thus the consensus is similar to the individual results (what is so expected). For the OAEI matchers, the consensus's results is in the range of the values of the best matchers, improving the results of the worse matchers. However, the cooperative model are able to filter out some false negative mappings while retrieving true positive mappings, slightly improving precision and recall of all matchers (i.e., negotiation and argumentation, respectively). For the *real* cases, in terms of F–measure, negotiation slightly outperforms the best matcher.

In the *library* case, cooperative models have showed to be promising, improving the worse matchers. Moreover, an important issue when using such dataset is the possibility to evaluate the four kinds of relations (i.e., *exactMatch*, *narrowMatch*, *broadMatch*, and *related match*) retrieved by the proposed matchers. In the OAEI tracks, such as anatomy and library, it is proved to be useful. Evaluating the *syntactic* matcher against the other matchers, it has achieved significant better results than matchers using, for example, instances of the ontologies (Co-occurrence and DSSim matchers).

6.4 Summary

In this chapter, the evaluation of the negotiation and argumentation models was presented. Two data sets are used: (i) the OAEI benchmark of ontologies in the bibliographic domain; and (ii) the real-world library case, from National Library of the Netherlands. Two groups of matchers were used: the proposed matchers and the OAEI 2007 participants. While the proposed matchers represent well-specific matching approaches, the OAEI matchers represent hybrid solutions.

For the OAEI benchmark, two experiment settings were specified: (i) using results from the proposed agents; and (ii) results from the systems that had participated in the OAEI 2007 benchmark. Negotiation and argumentation performed well, with negotiation achieving more satisfactory results.

In summary, the consensus obtained by negotiation and argumentation is not exactly the improvement of all individual results, but it represent intermediary values which are closer to the best matcher. Due the fact that individual performances are not always available, using cooperative models allows to achieve an equilibrium in the results.

Chapter 7

Conclusions

Ontology matching is viewed as a promisor solution to the semantic heterogeneity, supporting interoperability between ontology-based systems. An important research issue in ontology matching is to find ways of choosing among many techniques and their variations, and then combining their results.

In this thesis, the problem of combining different matching approaches was formalized using a cooperative agent-based approach. Specifically, two models were proposed: a negotiation model based on voting, and an argumentation model based on strength. For both models, the mappings are computed by agents using different matching approaches. For the former, the consensus between the matchers is represented by the number of supporters of a positive or negative mapping, where the greater number is choose as consensus.

The argumentation model was based on the Value-based Argumentation Framework (VAF). Such framework was extended in order to represent arguments with *confidence degrees*, according to the similarity degree between the entities being mapped. A new notion of argument acceptability was defined, which combines *values* (related with the agent's preference) and *strength* (confidence degree of an argument). Based on their preferences and degree of confidences, the matchers compute their preferred mapping sets. The arguments in such preferred sets are viewed as the set of globally acceptable arguments.

Using argumentation, it is possible to use the *values* to represent preferences between the matchers. Each approach represents a *value* and each agent represents an audience, with preferences between the *values*. The *values* are used to determine the preference between the different matchers. Moreover, each agent generates arguments with a ***strength***, based on the confidence returned by the matching technique. When the performance of the individual matchers are available, a complete preference order can be defined (i.e., A > B > C), while when this information is not available, a partial preference order must be specified (i.e., A > B; and A > C).

Differently from the argumentation model, in the negotiation a matcher that not performs well has the same weight in the voting than a matcher that performs very well. A simple relation of preference between the matchers is used only in the case of equals number of votes for positive and negative mappings, where the matcher with best performance decides the impasse (when such information is not available, a arbitrary choosing must be done).

A potential advantage of using argumentation is the possibility of adjust the preference between the matchers. On the other hand, the performance of the argumentation is related with the strength attributed to the mappings. It is recognized the importance of using arguments with strength, reflecting the confidence the matcher has in the similarity between two entities (the matching tools actually output mappings with a confidence measure). Such confidence levels are usually derived from similarity assessments made during the matching process, e.g. from edit distance measure between labels, or overlap measure between instance sets. However, there is no objective theory nor even informal guidelines for determining such confidence levels. Using them to compare results from different matchers is therefore questionable especially because of potential scale mismatches. For example, a same strength of 0.8 may not correspond to the same level of confidence for two different matchers. Moreover, using different values to represent the strength of counter-arguments of a positive mappings is a trade-off between precision and recall. Such evidence point to the necessity of a more comprehensive study in order to specify strengths that could balance these results.

A potential weakness of the argumentation model is related to the fact that an argument against a mapping can successfully attack all the arguments in favor of it, even if there are dozens of these (specially when using high strength for arguments representing negative mappings). For example, three arguments for a true positive mapping can be successfully attacked by just one argument representing a false negative mapping. On the other hand, as already commented, such fact can improve significantly the precision, while reducing the values of recall. As the OAEI 2006 Food track campaign (Euzenat et al. [2006]) – have indeed shown that the more often a mapping is agreed on, the more chances for it to be valid.

In average, negotiation performs better than argumentation (specially taking into account the values of F–measure). When considering the union of all mappers, both models had presented promising results. Negotiation is in average better, while the results of argumentation are directly related with the strength of the negative mappings (often improving the precision, often improving the recall).

It is hard to improve the best matcher, specially when there is a great intersection between the individual results. In this case, the performance of the cooperative models is similar to the results of the individual matchers. On the other hand, when the sets are disjunctive, the cooperative models are promising. Negotiation had proved to be useful due the fact that is based on how often a mapping is agreed on, augmenting the chances for it to be valid (again, as the OAEI 2006 Food track campaign, it has indeed shown that the more often a mapping is agreed on, the more chances for it to be valid). When using argumentation with

high strength for counter-arguments in negative mappings, significant values of precision are achieved. Thus, a combination of both approaches could be interesting, for instance adapting the S-VAF by introducing voting into the definition of successful attacks; by introducing weighted strength based on the individual performance of the matchers (what is not always available); or using a meta-matcher that combine both models.

Due the fact that ontology matching is a relative new research area, the data sets available for evaluation are still under construction and they are not very robust.

When analyzing the performance of the matchers, as the results of OAEI campaigns, Falcon had performed well in the majority of the tests. Specifically for the *benchmark* track, RiMON is in set of better results both in 2006 and 2007. However, similar techniques and strategies, when applied to different ontologies can produce different results. For instance, for *directory* and *anatomy* tracks, OLA and SAMBO performed better than the other participants. Basically, systems that use both ontology and data level information performed well. For complex ontologies, such as the OWL-DL ontologies of the *anatomy*, systems using specific domain knowledge show considerable improvement in the results. Few systems use promising techniques such as reasoning/inconsistency detection, or techniques based on constraint satisfaction problem (CSP), due to the characteristics of ontology itself and its representations. Moreover, reuse of mappings is also unexploited.

As future work, an argumentation framework based on voting could be specified, which considers the number of arguments in favor and against a mapping. As observed in the evaluation process, how often a mapping is agreed on, augmenting the chances for it to be valid. Second, a quantitative study about the use of different strengths for the arguments must be done, in order to evaluate the equivalences between the confidences returned by different matchers. Third, a meta-learner to combine the results of negotiation and argumentation could be specified. Fourth, other data sets could be exploited, in order to evaluate the quality of the semantic relations retrieved by the proposed matchers.

Appendix

Appendix A

Ontology Matching Systems

A.1 Ontology-level information

A.1.1 (Hovy, 1998)

(Hovy [1998]) describes two sets of heuristics for alignment and integration of ontologies: (i) *text matchers*; and (ii) *hierarchy matchers*. The *text matcher* includes concept names and definitions matchers. The name matcher compares the names of two concepts, where composite words are split into separated words and the maximum score is returned (*namescore* = square of number of letter matched + 20 points if words are exactly equal or 10 points if end of match coincides). The definition matcher compares definitions of the concepts. First, the definitions are separated into individual words and stop words are removed. With the remaining words, three values are computed: strength (ratio of number of words shared in both definition to number of words in the shorter definition), reliability (number of shared words), *defscore* (strength * reliability). The hierarchical matcher traverses the taxonomy in both super-concepts and sub-concepts directions (all concepts in the target ontology that are closer than 10 links to the considered concept). This verification produces the score *taxscore* = 1 \ number-of-links. The final score is given by the combination formula: *score* = *sqrt(namescore)* * defscore * (*10* * taxscore).

A.1.2 (Milo and Zohar, 1998)

(Milo and Zohar [1998]) propose a mechanism for data translation between input schemas, where the alignment is used for translating data instances of the source schema to instances of the target schema. The matching consists of basic functions, such as *match* and *descendent functions. The match function examines the labeling of the vertices of the two graphs and*

determines if they possibly match (using a dictionary to detect synonyms). The match is conditional on the matching of the components of the vertices (i.e. their descendants in the schema graph) as determined by the second function. For each pair of vertices of the input and output schemas, the function *descendent* returns two sets of descendants that need to be matched in order the two vertices to match (sets of direct children of the two vertices). The functions are combined sequentially based on their priorities.

A.1.3 SKAT (1999)

SKAT (Semantic Knowledge Articulation Tool) (Mitra et al. [1999]) is a rule-based system that semi-automatically discovers mappings between two ontologies. The ontologies are encoded as graphs, while the rules are provided by domain experts and encoded in first order logic. Matchers are applied sequentially, considering string-based matching and structure matching (considering nodes near the root of the first ontology against nodes near the root of the second ontology). The successor of SKAT is ONION (ONtology compositION) (Mitra et al. [2000]), which adds new matchers. The new linguistic matchers use word similarity table generated by a thesaurus or corpus-based matcher (the similarity score between two concepts is the average of the similarity scores of all possible pairs of words in their names). Structural matching is based on the results of the linguistic matching, looking for structural isomorphism between subgraphs of the ontologies. The structural matcher tries to match only the unmatched pairs from the linguistic matching.

A.1.4 DIKE (2000)

(Palopoli et al. [2000], Palopoli et al. [2003]) present DIKE (Database Intensional Knowledge Extractor) a matching tool for inferring semantics relations among schema objects from different database schemas. The matching process is based on *terminological* and *structural* proprieties. The *terminological* matchers include analysis of synonymous and homonymous (using external resources, such as WordNet). The *structural* matchers apply subschema similarities, such as similarities between schemas fragments. A value in the range [0,1] is associated to each similarity. The algorithm works computing sequentially the similarities. A final coefficient is produced, taking into account the different proprieties.

A.1.5 Artemis (2001)

Artemis (Analysis of Requirements: Tool Environment for Multiple Information Systems) (Castano et al. [2001]) performs affinity-based analysis and hierarchical clustering of database schema elements. Affinity-based analysis represents the matching step. In a sequential man-

ner, it calculates the name, structural and global affinity coefficients exploiting a common thesaurus, which is built with the help of WordNet or manual input. Based on global affinity coefficients, a hierarchical clustering technique categorizes classes into groups. For each cluster, it creates a set of global attributes and the global class. Logical correspondences between the attributes of a global class and source schema attributes are determined through a mapping table.

A.1.6 Anchor-Prompt (2001)

Anchor-Prompt (Noy and Musen [2001]) is an ontology alignment and merging tool. It is a sequential matching algorithm that takes as input two ontologies, internally represented as graphs (classes are nodes and slots are links), and a set of anchors – pairs of related terms defined by the user or automatically identified by string-based techniques. From this set of previously identified anchors, Anchor-PROMPT produces a set of new pairs of semantically close terms. To do that, Anchor-PROMPT traverses the paths between the anchors in the corresponding ontologies. A path follows the links between classes defined by the hierarchical relations or by slots and their domains and ranges. Anchor-PROMPT then compares the terms along these paths to find similar terms. If if two pairs of terms from the source ontologies are similar and there are paths connecting the terms, then the elements in those paths are often similar as well.

A.1.7 OntoBuilder (2001)

OntoBuilder (Modica et al. [2001], Gal et al. [2004]) project supports the extraction of ontologies from Web search interfaces, that enables fully-automatic ontology matching. It operates in two phases: (i) ontology creation and (ii) ontology adaptation. In the first phase, an initial ontology is created by extracting it from web sites. The adaptation phase includes on-the-fly match and interactive merge of related ontologies with the initial ontology. In this phase, the users suggest the web sites to explore, which are used to ontology extraction. This result in a candidate ontology, which is merged into the initial one. This process involves a matching process, which consists in the sequential execution of several matchers (that includes a preprocessing to remove stop terms and noisy characters): substring matching, thesaurus lookup, and manual matching.

A.1.8 Cupid (2001)

Cupid (Madhavan et al. [2001]) is a matching system based on *linguistic* and *structural* approaches. First, it matches individual schema elements based on their names, data types,

domains, etc. A thesaurus is used to help match names by identifying short-forms, acronyms, and synonyms. The result is a linguistic similarity, *lsim*, between each pair of elements. Next, structural matching of schema elements is made. The structural match depends in part on linguistic matches calculated initially, resulting in a structural similarity coefficient, *ssim*. The weighted similarity (*wsim*) is a mean of *lsim* and *ssim*: $wsim = wstruct \times ssim + (1-wstruct) \times lsim$, where the constant *wstruct* is in the range 0 to 1.

A.1.9 COMA (2002)

COMA (COmbination of MAtching algorithms) (Do and Rahm [2002]) is a matching tool based on parallel composition of matchers. It provides a library of matching algorithms, a framework to combine the results, and a platform for the evaluation of the different matchers. It contains 6 elementary matchers, 5 hybrid matchers, and 1 reuse-oriented matcher. They exploit different kinds of schema information, such as names, data types, and structural properties, or auxiliary information, such as synonym tables and previous match results. The elementary matchers implement, basically, string-based techniques, such as affix, n-gram, edit distance. The hybrid matchers are based on thesauri lookup. The reuse-oriented matcher uses previously obtained results for new schemas. The match result is a set of mapping elements specifying the matching schema elements together with a similarity $\in$ [0,1] indicating the plausibility of their correspondence. A combined match result from the individual matcher results is stored in a similarity cube. This is achieved in two sub-steps: aggregation of matcher specific results and selection of mapping candidates. First, for each combination of schema elements the matcher-specific similarity values are aggregated into a combined similarity value, e.g. by taking the average or maximum value. Second, a selection strategy is applied to choose the mapping candidates for a schema element, e.g. by selecting the elements of the other schema with the best similarity value exceeding a certain threshold.

A.1.10 Similarity flooding (2002)

Similarity flooding (Melnik et al. [2002]) is a structural algorithm that can be used for matching of data schemas. The elements of the schemas represent artifacts like relational tables and columns, or products and customers. The algorithm is based on the following steps. First, it converts the models to be matched into directed labeled graphs. These graphs are used in an iterative fixpoint computation whose results indicate what nodes in one graph are similar to nodes in the second graph (using string based comparison, such as common prefix and suffix). For computing the similarities, it relies on the intuition that elements of two distinct models are similar when their adjacent elements are similar (i.e., a part of the similarity of two elements propagates to their respective neighbours).

A.1.11 XClust (2002)

XClust (Lee et al. [2002]) is a tool for integrating multiple DTDs, using strategies based on clustering. The system works in two phases: (i) DTD similarity computation; and (ii) DTD clustering. In the first phase, pairwise similarities between labeled trees are computed, using matchers that exploit schema names and structural information. Schema names matchers compute the similarity using comparison of synonymous from WordNet and a cardinality restrictions that considers a table compatibility in order to compute the cardinality similarity. Structural matchers exploit previously name similarity and are based on (a) similarity of paths – which is computed as a normalized sum of name similarity between the sets of elements of these paths (elements from root to the node under consideration); (b) similarity of descendants; and (c) similarity of leaves. The structural similarities are aggregated as a weighted sum. The result of the first phase is the similarity matrix of a set o DTDs. In the second phase, based on this matrix, a hierarchical clustering is applied to group DTDs into clusters.

A.1.12 S-Match (2004)

S-Match (Giunchiglia et al. [2004a]) adopts the idea of semantic matching, meaning the use of logical relations. It takes as input two graph-like structures (classifications, XML schemas, ontologies) and returns as output logic relations (equivalence, subsumption) between the nodes of the graphs. The relations are determined by (i) expressing the entities of the ontologies as logical formulas; and (ii) reducing the matching problem to a propositional validity problem. The entities are translated into propositional formulas which express the concept description as encoded in the ontology structure and in external resources as WordNet. The algorithm works in a sequential system with parallel composition at the element level. The input ontologies are preprocessed and the algorithm is based on two main steps. First, the synonymous terms are captured using WordNet (element level) and several basic matchers such as n-gram and edit-distance can be used. Second, the structural schema properties are taken into account, where the path to the root is computed (structural level). The structural matcher includes the SAT solvers. Element level matchers provide the input to the structural level matcher, which is applied on to produce the set of semantic relations between concepts as the matching result.

A.1.13 MoA (2005)

MoA (Kim et al. [2005]) in an ontology matching and merging tool that comprises (i) a library of methods for importing, matching, modifying, and merging OWL ontologies; and (ii) a shell for using these methods. The matching approach is based on concept similarity

derived from linguistic rules. It is a sequential solution, involving a preprocessing phase. This phase includes three steps: (a) names of classes and properties are tokenised; (b) tokens of entities are associated with their meaning by using the WordNet senses; (c) meanings of tokens of ancestors of the entity are taken into account. The matching is based on rules, i.e., equivalence between two classes or proprieties holds when there is equivalence between these entities in the steps (b) and (c).

A.1.14 ASCO (2005)

ASCO (Bach et al. [2004], Bach and Dieng-Kuntz [2005]) is a system to match ontologies described in OWL or RDF. The matching is sequential and has three steps. First, the terms and expressions are normalized (for instance, punctuation, upper case, special symbols) and different string comparisons (edit distance) and external resources (WordNet) are used. Based on token similarities, the similarity between sets of tokens is computed using TFIDF. The obtained values are aggregated through a weighted sum. Second, structure matching is computed by propagating the input of linguistic similarities. It is an iterative fixed point computation algorithm that propagates similarity to the neighbors (subclasses, superclasses, and siblings). The propagation terminates when the class and relation similarities do not change after an iteration or a certain number od iterations is reached. Third, linguistic and structural similarities are aggregated in a weighted sum.

A.1.15 OMEN (2005)

OMEN (Ontology Mapping ENhancer) (Mitra et al. [2005]) is a framework to improve existing ontology mappings using a Bayesian Network. It applies a set of meta-rules that capture the influence of the ontology structure and the semantics of ontology relations and matches nodes that are neighbors of already matched nodes in the two ontologies. It takes as input two ontologies, an initial probability distribution derived from element level linguistic matchers, and positive and negative evidence thresholds. Also, it provides a structural level matching algorithm, by deriving the new mappings or discarding the existing false mappings. The following summarizes the Omen algorithm: (i) if initial probability of a match is above a given threshold, create a node representing the match and mark it as evidence node; (ii) for each pair of concept pair of concepts, create a node in the BayesNet graph; (iii) create edges between the added nodes using the rules for top-down or bottom-up iterations; (iv) prune out nodes that are at a distance greater than k from an evidence node (a node with a priori probability above the positive threshold or below the negative threshold); (v) use the meta-rules to generate Conditional Probability Tables (CPT) for the BayesNet; (vi) run the BayesNet to generate the a posteriori probabilities of each node; (vii) select those nodes with a posteriori probabilities over a given threshold as matches. The output of this process is a

new set of matches.

A.1.16 H-Match (2006)

H-Match (Castano et al. [2006]) is a system for matching distributed OWL ontologies. It is based on linguistic (based on WordNet) and structural matching techniques for the evaluation of affinity considering concept names and concept contexts. These are combined by using weighting schemas (linear combination), resulting a *semantic affinity* final measure in the range [0, 1]. The linguistic techniques consider names of ontology elements and their meaning. To capture the meaning of names, a thesaurus of terms and weighted terminological relationships are exploited. In particular, it extends the Artemis thesaurus-based approach for name affinity management by extending the matching linguistic features (such as automatic handler of composite terms) of ontology elements in order to rely only on the WordNet lexical system, providing a fully-automated matching process. The contextual affinity considers neighbors concepts e.g., linked via taxonomical or mereological (parf-of) relations of the actual concept. Moreover, another feature of H-MATCH is that it can be dynamically configured for adaptation to the semantic complexity of the ontologies to be compared, where the number and type of ontology features that can be exploited during the matching process is not known in advance. This is achieved by means of four matching models: *surface*, *shallow*, *deep*, and *intensive*. Computation of linguistic affinity is a common part of all the matching models. In surface model, only the linguistic affinity is considered. The other three models take into account various contextual features. The shallow model considers concept proprieties, while the deep and the intensive models extend previous models by including relations and property values, respectively.

A.1.17 MapOnto (2006)

MapOnto (An et al. [2006]) is a semi-automatic tool that assists users to discover semantic relationships between a database schema (relational or XML) and ontologies. The system takes as input three elements: (i) an ontology specified in an ontology representation language (e.g., OWL), (ii) relational or XML schemas; and (iii) simple correspondences (e.g. between XML attributes and ontology datatype properties). Input schema and ontology are internally encoded as labelled graphs. The system looks for connections among the graphs, producing in a semi-automatic way a set of complex mappings formulas expressed in a subset of first-order logic. The list of logical formulas is then ordered and the mappings can be selected by the user.

A.1.18 HCONE (2006)

HCONE (Kotis et al. [2005]) is a tool for ontology matching and merging, which explores different levels of interaction with users. Initially, an alignment between the input ontologies is computed, using WordNet. Next, the alignment is processed by using merging rules and a new ontology is created. The basic matcher works as following: all WordNet senses for a given concept are obtained; the hypernym and hyponym relations are retrieved from the senses; an association matrix relating the n most frequently occurring terms in the m senses is created; using the matrix, Latent Semantic Indexing is applied to compute the grades for what is the correct WordNet sense (i.e., the most plausible meaning for the concepts under consideration). Finally, the relationships between the concepts are computed: equivalence between two concepts holds if the same WordNet sense has been chosen for those concepts; subsumption relation is returned if a hypernym relation holds between the WordNet senses corresponding to these concepts. Moreover, different level of user interaction are considered: fully automated, semi-automated, and user-based. Users can provide feedback on what is to be the correct WordNet sense (user-based) or in some limited cases, by exploiting heuristics (semi-automated).

A.1.19 DSSim (2007)

DSSim (Nagy et al. [2007]) is an ontology mapping system that is used with a multi agent ontology mapping framework in the context of question answering. It works as follows: (i) it takes a concept (or property) from ontology 1 and consider it as the query fragment that would normally be posed by a user. From the query fragment, a graph which contains the close context of the query fragment such as the concept and its properties is built; (ii) it takes syntactically similar concepts and properties (using string-based techniques to match names and name descriptions, such as edit distance and Jaccard similarity) and its synonyms (using the WordNet) to the query graph from ontology 2 and builds a graph that contains both concepts (properties) and its synonyms; (iii) different similarity algorithms (considered as different experts in evidence theory) are used to assess quantitative similarity values (converted into belief mass function) between the nodes of the query and ontology fragment which is considered as an uncertain and objective assessment. Then the information produced by the different algorithms is combined using the Dempster's rule of combination; (iv) based on the combined evidences it is assessed the semantic similarity (based on the analysis of their positions within the graphs) between the query and ontology graph fragment structures and select those in which we calculate the highest belief function; (v) the selected concepts are added into the alignment.

A.1.20 OntoDNA (2007)

OntoDNA (Kiu and Lee [2006], Kiu and Lee [2007]) is an ontology matching and merging system based on hybrid unsupervised clustering methods, namely Formal Concept Analysis (FCA), Self-Organizing Map (SOM) and K-means incorporated with lexical similarity. Basically, OntoDNA relies on ontological concepts and properties name for mapping and merging ontologies. It comprises four steps: (i) using FCA to capture the properties and structural relationships among the concepts (i.e., discovering conceptual pattern between the ontologies); (ii) pre-linguistic processing is applied to normalize the attributes and a set of mapping rules is applied to reconcile intents in these attributes (the reconciled formal context is used as input for semantic similarity discovery in the next step); (iii) SOM and k-means are applied for semantic similarity mapping based on the conceptual pattern discovered in the formal context; and (iv) the mapping rules are applied to discover semantic similarity between ontological concepts in the clusters. The ontological concepts of the target ontology are updated to the source ontology based on merging rules.

A.1.21 PriorPlus (2007)

The PriorPlus (Mao and Peng [2007]) is an extension of the Prior (Mao and Peng [2006], Mao et al. [2007]) system. In addition to the profile similarity and the edit distance of name of elements used in the Prior, the PriorPlus considers structure similarity as well and adaptively aggregate different similarities based on their harmony (i.e., a term used to represent the similarity between ontologies). Moreover, it has a brand new Neural Network-based Constraint Satisfaction Solver. It works in three sequential phases. First, both linguistic and structural similarities (phase i) are computed in parallel. Linguistic similarity involves calculating the profile similarity and the edit distance of elements' name (i.e. the profile of a concept is a combination of all linguistic information of the concept – the concept's name + label + comment + property restriction + other descriptive information). To calculate the structure similarity of two elements, various structural features are extracted, e.g. the number of its sub-elements, the number of its direct property, the depth of the element to the root etc. The difference between these structural features are calculated and normalized to represent its structure similarity. The outputs of the similarity generation process are three similarity matrixes (name-based, profiles, and structural similarities), where each matrix denotes a kind of similarity of two ontologies. In the second phase (phase ii), these matrixes are used to calculate three types of harmony of ontologies: name harmony, profile harmony and structure harmony. Basically, each harmony is computed by the number of cells that own the highest similarity in its corresponding row/column divided by the number of elements in both two ontologies. The final harmony corresponds to a weighted aggregation of the the three harmonies. If the final harmony is greater than a threshold, the Neural Network-based Constraint Satisfaction Solver is used to refine the previous results (phase iii).

A.1.22 TaxoMap (2007)

TaxoMap (Zargayouna et al. [2007]) is an approach to align taxonomies which relies on terminological techniques applied sequentially. It proposes essentially subclass relation mappings. Only concepts that have strictly the same label are matched with an equivalence relation. The remaining concepts of the source ontology are matched with a subclass relation which denotes a proximity relation. The alignment techniques are based on the Lin's similarity measure, which compares strings and has been adapted to take into account the importance of words inside expressions. The terminological techniques are performed in three sequential steps: (i) search for equivalents, where concepts with strong similarity (greater than a threshold) are selected; (ii) labels inclusion, where the inclusions of name strings between the two labels are proposed as a subclass mapping; (iii) relative similarity, where if the name string of the concepts (c_1 and c_2) with the higher similarity measure is not included in the name string of c_2, but if its similarity measure is significantly highest than the measure of the others, c_2 is considered as a brother of c_1 and the system proposes a subclass relationship between c_1 and the father node of c_2.

A.1.23 X-SOM (2007)

X-SOM – eXtensible Smart Ontology Mapper – (Curino et al. [2007a], Curino et al. [2007b]) is an extensible ontology matcher that combines various matching algorithms by means of a feed-forward neural network. It is composed by three subsystems: Matching, Mapping and Inconsistency Resolution. The Matching Subsystem is constituted by an extensible set of matching modules, each of which implements a matching technique (i.e., linguistic, which is based on the edit distance and search in the WordNet; structural technique, which uses the Graph Matching for Ontologies algorithm; and semantic technique, which is based on the use of use background, contextual and prior knowledge to compute the similarity degree between two resources). Each module produces a so-called similarity map, and the several maps are then combined by means of a feed-forward neural network in order to produce an aggregated similarity degree. Given these aggregate matching values, the Mapping Subsystem computes a set of candidate mappings by applying, to the set of matchings, a pair of configurable threshold values. The first threshold is called discard threshold; the matchings with a similarity degree lower than it are discarded a-priori. The second threshold is called accept threshold, and the matchings with a similarity degree greater than it are accepted as candidate mappings. The remaining matchings, whose similarity is between the two thresholds, are considered as uncertain and manually evaluated by the user. Finally, the Inconsistency Resolution module takes as input the candidate mappings from the Mapping Subsystem and produces a set of mappings, in which at least all the logical inconsistencies have been solved.

A.2 Data-level information

A.2.1 CAIMAN (2001)

CAIMAN (Lacher and Groh [2001]) is a system that facilitates the exchange of relevant documents between different communities of interest. It is assumed that each community organizes its documents according to its own categorization scheme (ontology). It exploits this ontology for information retrieval, where related documents are retrieved on a concept granularity level from a central community document repository. To find the related concepts in the queried ontology, CAIMAN performs an ontology matching. The main idea of the matching algorithm is to calculate a probability measure between the concepts of two ontologies by applying machine learning techniques for text classification. Based on documents, a representative feature vector (a word count, weighted by TFIDF) is created for each concept. The cosine measure is computed for two of those class vectors and by using thresholds, the resulting alignment is produced.

A.2.2 FCA-Merge (2001)

FCA-Merge (Stumme and Maedche [2001]) is a system for merging ontologies following a bottom-up approach which offers a structural description of the merging process. The method is guided by application-specific instances of the given source ontologies, that are to be merged. It is based on techniques from natural language processing and Formal Concept Analysis (FCA) to derive a lattice of concepts as a structural result of FCA-Merge. The process of merging consists of three steps: (i) instance extraction; (ii) concept lattice computation; and (iii) interactive generation of the final merged ontology. In the first phase, the set of instances is extracted from text documents. In the second phase, the system uses formal concept analysis techniques in order to compute the concept lattice involving both ontologies. The last step consists of deriving the merged ontology from the concept lattice. The produced lattice is explored and transformed by users who further simplify it and generate the taxonomy of an ontology. The result is a merge rather than an alignment. However, the concepts that are merged can be considered as exactly matched.

A.2.3 LSD (2001)

LSD (Learning Source Description) (Doan et al. [2001]) is a system for semi-automatic discovery of one-to-one alignments between the elements of source schemas and a mediated schema in data integration. The main idea is to learn from mappings created manually between the mediated schema and some of the source schemas, in order to propose in an

automatic manner the mappings for subsequent source schemas. The system consists of four components: *base learner*, *meta-learner*, *prediction converter*, and *constraint handler*. It operates in two phases: training and matching. In the training phase, LSD first asks the user to manually specify the mappings for several sources. Second, it extracts some data from each source, creating the training examples for the base learners from the extracted data. Different base learners require different sets of training examples. Next, it trains each base learner on the training examples. Finally, it trains the meta-learner. The output of the training phase is the internal classification models of the base-learner and meta-learner. In the matching phase, the trained learners are used to match new source schemas. First, LSD extracts some data from the source and creates for each source-schema element a column of elements that belongs to it. Second, it applies the base learners to the elements in the column, then combines the learner's predictions using the meta-learner and the prediction converter. Finally, the constraint handler takes the predictions, together with the available domain constraints, and outputs 1-1 mappings for the target schema. The user can either accept the mappings or provide some feedback and ask the constraint handler to come up with a new set of mappings.

A.2.4 AutoMatch (2002)

AutoMatch (Berlin and Motro [2002]) is a schema matching system that uses machine learning techniques to automated discovery of mappings between attributes of database schemas. Based primarily on Bayesian learning, the system acquires probabilistic knowledge from examples that have been provided by domain experts. This knowledge is stored in a knowledge base called the attribute dictionary. When presented with a pair of new schemas that need to be matched (and their corresponding database instances), Automatch uses the attribute dictionary to find an optimal matching. This dictionary characterizes different attributes by means of their possible values and the probability estimates of these values. Furthermore, the dictionary may be extended to contain any attribute metadata that has a probabilistic interpretation (e.g. attribute names or string patterns).

A.2.5 (Kang and Naughton, 2003)

An instance-based approach for discovering correspondences between attributes of relational schemas is proposed by (Kang and Naughton [2003]). It is a two-step technique that works even in the presence of opaque column names and data values. In the first step, it measures the pair-wise attribute correlations in the tables to be matched and construct a dependency graph using mutual information as a measure of the dependency between attributes. Two table instances are taken as input and the corresponding dependency graphs are constructed based on the *minimal information and entropy*. Mutual information measures the reduction

in uncertainty of one attribute due to knowledge of the other attribute (the amount the information captured in one attribute about the other). It is zero when two attributes are independent and increases as the dependency between the two attributes grows. Mutual information is computed over all pairs of attributes in a table. In dependency graphs, a weight on an edge stands for mutual information between two adjacent attributes. A weight on a node stands for entropy of the attribute. In the second stage, it finds matching node pairs in the dependency graphs by running a graph matching algorithm (matching node pairs are discovered between the dependency graphs). The quality of matching is assessed by using metrics such as Euclidean distance. The distance is assigned to each potential correspondence between attributes of two schemas and one-to-one alignment which is a minimum weighted graph matching.

A.2.6 GLUE (2004)

GLUE (Doan et al. [2004]) is a machine learning matching system that follows a multi-strategy learning approach, involving several basic matchers and a meta-matcher. The system works in three steps. First it learns the joint probability distribution of classes of two taxonomies, exploiting two basic matchers (content learner using naive Bayes technique and the name learner). The meta-learner performs a linear combination of the basic matchers. Weights for these matchers are assigned manually. In the second step, the system estimates the similarity between two classes in a user-supplied function of their joint probability distributions. This result in a similarity matrix between terms of two taxonomies. Finally, some domain dependent (subsumption) and domain-independent (if all children of the node x match node y, then x also matches y), constraints (heuristics) are applied by using a relaxation labeling technique. They are used in order to filter some of the matches out of the similarity matrix and keep only the best ones.

A.2.7 iMAP (2004)

iMAP (Dhamankar et al. [2004]) is a system which semi-automatically discovers both one-to-one (address = location) and complex matches (address = concat(city,state)) between database schemas. It reformulates schema matching as a search in a match space. To search, iMAP employs a set of searchers, each discovering specific types of complex matches. It exploits a variety of domain knowledge, including past complex matches, domain integrity constraints, and overlap data. For instance, a text searcher considers the concatenation of text attributes, while numeric searcher considers combining attributes with arithmetic expressions. Specifically, iMAP works in three steps. First, matching candidates are generated by applying basic matchers. During the second step, for each target attribute, matching candidates of the source schema are evaluated by exploiting additional types of information

(naive Bayes evaluator, for instance), resulting additional scores. All the score are combined into a final one, resulting in a similarity matrix between pairs (target attribute, match candidate). Finally, by using a set of domain constraints and mappings from the previous match operations, the similarity matrix is cleaned up such that only the best matches for target attributes are returned as the result.

A.2.8 SBI (2004)

SBI (Similarity Based Integration) (Ichise et al. [2004]) is a system for automatic statistical matching among classifications. The system aims to determine correspondences between classes of two classifications by statistically comparing the membership of the documents of these classes. The pair of similar classes are determined in a top-down fashion by using the k-statistic method. These pairs are considered to be the final alignment.

A.2.9 DUMAS (2005)

DUMAS (DUplicate-base MAtching of Schemas) (Bilke and Naumann [2005]) is an instance-based approach which identifies one-to-one alignments between relational schemas. It performs horizontal matching, traversing tables in search for similar rows (or tuples), in effect detecting duplicates. Once a few duplicates have been discovered, deriving a schema matching is simple in principle: same or similar data values among the duplicates imply corresponding attributes of the schemas. This approach solve two problems: (i) detecting duplicates among databases with opaque schemas and (ii) deriving a schema matching from a set of fuzzy duplicates. For the search, tuples are viewed as strings and a string comparison is used to compare two tuples. Specifically, tuples are tokenized and each token is assigned a weight based on TFIDF scheme. The algorithm ranks tuple pairs according to their similarity and identifies the k most similar tuple pairs. Next, based on the k duplicate pairs with highest confidence, the correspondences between attributes are derived (i.e., if two field values are similar, then their respective attributes match). A field-wise similarity comparison is made for each k duplicates, resulting in a similarity matrix. For comparing tuple fields, a variation of a TFIDF-based measure is used, allowing the consideration of similar terms as opposed to equal terms. The resulting alignment is extracted from the similarity matrix by finding the maximum weight matching.

A.2.10 sPLMap (2005)

sPLMap (Probabilistic Logic-based Mapping) (Nottelmann and Straccia [2005]) is a formal framework for learning mappings between heterogeneous schemas which is based on

logics and probability theory. It combines different classifiers for finding suitable mapping candidates (together with their weights), and selects that set of mapping rules which is the most likely one. sPLMap operates in three main phases. First, it evaluates the quality of all possible individual correspondences on the basis of probability distributions, selecting the set of correspondences that maximizes probability on the basis of instance data. Then, for each correspondence, matchers are used as quality estimators: they provide a measure of the plausibility of the correspondence. The following matchers have been used: (i) same attribute names, (ii) exact tuples, (iii) the k neighbor classifier, and (iv) the naive Bayesian classifier. The result of these matchers are aggregated by linear or logistic functions, or their combinations.

A.2.11 (Isaac et. al., 2007)

An instance-based matcher (Isaac et al. [2008b]), developed in the context of the STITCH project[1], makes use of the individual objects described by the concepts that are to be mapped. The instances of a concept c, will be the set of objects—*e.g.*, books—which are related to this concept via a `subject` annotation property. This matcher considers the most elementary form of extension for a concept: the set of its instances. For each candidate mapping between two concepts, the tool accesses their instance sets and measure their overlap, which is considered as the confidence level for an equivalence *exactMatch* relation. The measure is adapted from the standard Jaccard similarity, so that it assigns a smaller score to less frequently (co-)occurring concepts. Evidence for concepts that have just one co-occurring instance is weighed as much as mapping two concepts would get when a large number of concepts have 20% in their intersection.

A.3 Ontology and data level information

A.3.1 SEMINT (2000)

SEMINT (SEMantic INTegrator) (Li and Clifton [2000]) is a system based on neural networks to identify correspondences between attributes in heterogeneous databases. It applies both schema and instance level information to produce rules for matching corresponding attributes. First, it extracts from two databases all necessary information for matching. This includes normalized schema information (field specifications, such as datatypes length, constraints) and statistical about data values (character pattern, such as ratio of numerical characters, ratio of white spaces and numerical patterns, such as mean, variance, standard deviation). Second, by using a neural network as a classifier with self-organizing map

[1] http://www.cs.vu.nl/STITCH/

algorithm, it groups the attributes based on the similarity of the features for a single database (the first). Then, it uses a back-propagation neural network for learning and recognition. Based on the previously obtained clusters, the learning is performed. Using a trained neural network on the first database features and clusters, the system recognizes and computes similarities between the categories of attributes from the first database and the features of attributes from the second database. A list of match candidates is generated, which are to be confirmed or discarded by users.

A.3.2 Clio (2000)

Clio (Miller et al. [2000], Hernández et al. [2001], Haas et al. [2005]) is a system that provides a declarative way of specifying schema mappings between either XML or relational schemas. As a first step, the system transforms the input schemas into an internal representation. It combines in a sequential manner instance-based attribute classification via a variation of a naive Bayes classifier and string matching between element names by using an edit distance. Then, taking the value correspondences together with constraints coming from the input schemas, Clio compiles these into an internal query graph representation. In particular, an interpretation of the input correspondences is given. Thus, a set of logical mappings with formal semantics is produced.

A.3.3 IF-Map (2003)

IF-Map (Information Flow based Mapping) (Kalfoglou and Schorlemmer [2003a]) draws from the works on alignment of ontologies (Schorlemmer [2002]) and on the heuristics defined by (Kalfoglou [2000]), to analyze prospective mappings between ontologies. The method is based on the mathematical theory of information flow of (Barwise and Seligman [1997]), a general model that attempts to describe the information flow in distributed system. The matching is based on a reference ontology (global), which it is assumed to represent an agreed understanding between other (local) ontologies of the domain. When the reference ontology can be expressed in each local ontology and instances of the local ontologies can be associated to concepts in the reference ontology, the IF-Map applies formal concept analysis between the three ontologies to extract an alignment. It captures, by means of two pairs of contra-variant functions, an existing duality between concepts and instances: each pair consists of a map of concepts and map of instances, and pointing in the opposite direction. From a channel-theoretic perspective, sharing knowledge involves a flow of information that crucially depends on how the instances of different ontologies are connected together. If the mappings are not available, it generates candidate pairs of mappings and artificial instances. Such instances are generated via the enforcement of constraints which are induced by the definition of the reference ontology and by heuristics on string-based and structure-based

methods.

A.3.4 NON (2004)

NON (Naive Ontology Mapping) (Ehrig and Sure [2004]) and QOM (Quick Ontology Mapping) (Ehrig and Staab [2004]) are components of the FOAM framework. NON is based on a set of manually encoded mapping rules, and adopts the idea of parallel composition of matchers. It is able to discover one-to-one mappings between single entities. The system support 17 rules, which are based on various string-based (i.e., if labels are the same, the entities are probably also the same), structural techniques (i.e., if superconcepts are the same, the actual concepts are similar to each other), and instance-based techniques (i.e., instances that have the same mother concept are similar or if concepts have a similar low/high fraction of the instances, the concepts are similar). The combination of the matcher applies both manual and automatic approaches to learn how to combine the methods. It includes a summarizing over the n weighted similarity methods; a sigmoid function, which has to be shifted to fit our input range of [0 . . . 1] (a high similarity value should be weighted over-proportionally whereas a low value practically can be abandoned); and a machine learning technique to learn the best matcher.

A.3.5 QOM (2004)

QOM (Quick Ontology Mapping) is a variation of the NON system that analyzes the trade off between effectiveness (i.e. quality) and efficiency of the mapping generation algorithms. The approach is based on the idea that the loss of quality in matching algorithms is marginal while the improvement in efficiency can be significant. QOM is grounded on matching rules of NON, but for the purpose of efficiency, the use of some rules, such as the rules that traverse the taxonomy, are restricted. QOM avoids the complete pairwise comparison of trees in favor of an incomplete top-down strategy, focusing only on promising matching candidates. The similarity produced by basic matchers (matching rules) are refined by using a sigmoid function, thereby emphasizing high individual similarities and de-emphasizing low individual similarities. They are then aggregated through weighted average.

A.3.6 (Embley et. al., 2004)

A parallel composition approach to discover one-to-one, one-to-many, many-to-many correspondences between graph like structures is proposed by (Embley et al. [2004]). The matching is performed by a combination (an average function) of multiple matchers. The basic element level matchers include: (i) *name matcher*, which comprises string compar-

isons, linguistic normalization (stemming and removing stop words), and it detects synonymous among nodes names with the help of WordNet; (ii) *value-characteristic matcher*, which determines where two values of schema elements share similar value characteristics, such as means or variances of numerical data (based on instance data); and (iii) data-frame matcher, which finds instance-level mappings by applying data-frame recognizers to identify data values in object sets (it compares object values indirectly, i.e., if the same data frame recognizes values from two different object sets, there is a strong likelihood that these object sets match). For both, name and value-characteristic matchers, mapping rules are obtained training a C4.5 decision-tree generator over WordNet characteristics using synonym names found in a variety of database schemas and over value characteristics using sets of data values from a domain of interest, respectively. The structure level matchers are use to suggest new correspondences as well as to confirm correspondences identified by element level matchers. Examples of structural matchers include considering similarities between the neighbor elements computed by element level matchers and using an external domain ontology C and to match the schemas A and B to the C, in order to decide if A corresponds to B.

A.3.7 OLA (2004)

OLA (OWL Lite Aligner) (Euzenat and Valtchev [2004]) is an ontology matching system based on the analysis of classes, constraints, and data instances. It first compiles the input ontologies into graph structures, unveiling all relationships between entities. These graph structures produce the constraints for expressing a similarity between the elements of the ontologies. The similarity between nodes of the graphs follows two principles: (i) it depends on the category of node considered (class or property); and (ii) it takes into account all the features of this category (superclasses, properties). The distance between the nodes in the graph are expressed as a system of equations based on string-based, language-based, and structure-based similarities (as well as taking instances into account whenever necessary). These distances are almost linearly aggregated. For computing these distances the algorithm starts with base distances measures computed from labels and concrete datatypes. Then, it iterates a fixed point algorithm until no improvement is produced.

A.3.8 RiMON (2004)

RiMOM (Risk Minimisation based Ontology Mapping) (Tang et al. [2004]) approach formalizes ontologies as a decision making problem. Given two ontologies, it aims at an optimal and automatic discovery of alignments which can be complex (such as including concatenation operators). The approach firsts searches for concept-to-concept correspondences and then for property-property correspondences. The matching process has the following steps:

(i) *matcher selection* – if two ontologies have high label similarity, then RiMON will rely more on linguistic based strategies; while if the two ontologies have a high structure similarity factor, RiMON will exploit similarity propagation based strategies on them; (ii) *matcher execution* – matchers are executed independently, which can include linguistic normalization of labels, such tokenisation, expansion of abbreviations and acronyms, edit distance, matcher that looks for label similarity based on WordNet, k-nearest neighbors statistical learning, naive Bayes matcher, and other heuristics and taxonomic structure similarity. The result is a cube of similarity in [0,1] for each pair of entities from the two ontologies; (iii) *results combination* – the results are combined by aggregating the values produced during the previous step into a single value, by using linear interpolation; (iv) *similarity propagation* – if the two ontologies have high structure similarity factor, RiMON employs an algorithm called similarity propagation (using structural information) to refine the found alignments and to find new alignments that cannot be discovered using the other strategies; (v) *alignments extraction* – RiMON extracts alignment for a pair of ontologies based on thresholds and some refinement heuristics to eliminate unreasonable correspondences (for example, use concept-to-concept correspondences to refine property-to-property correspondences); and (vi) *iteration* – it iterates the above described process by taking the output of one iteration as input into the next iteration until no new correspondences are produced. At each iteration, users can select matchers, and approve and discard correspondences from the returned alignment.

A.3.9 oMap (2005)

oMap (Straccia and Troncy [2005]) is a general framework combining several specific classifiers, which use the semantics of the OWL axioms for establishing equivalence and subsumption relationships between the classes and the properties defined in the ontologies. The matchers are grouped into three groups: (i) a classifier based on string similarity measure used on entity names; (ii) a naive Bayes classifier used on instance data; and (iii) a formal matcher which propagates initial weights through the ontology constructors used in the definitions of ontologies entities. The semantic matcher takes as input an alignment associating plausibility of a new alignment by propagating these measures through the definitions of the considered entities. The propagation rules depend on the ontology constructions, when passing through a conjunction, the plausibility will be minimized. The matchers can be combined in parallel, in which case their results are aggregated through a weighted average, such that the weights correspond to the credit accorded to each of the classifiers; or in sequence, in which case each matcher only adds new correspondences to the input ontologies. A typical order is first strings similarity, before naive Bayes, and then the semantic matcher.

A.3.10 (Madhavan et. al., 2005)

An approach to schema matching that exploits domain specific knowledge via an external corpus of schemas and mapping as well as input information from schemas under consideration is proposed by (Madhavan et al. [2005]). The approach is inspired from the use of corpus in information retrieval, where similarity between queries and concepts is determined based on analyzing large corpora of text. In schema matching, such a corpus can be initialized with a small number of schemas obtained, for example, by using available standard schemas in the domain of interest, and should eventually evolve in time with new matching techniques. In particular, basic matchers include: (i) name learner; (ii) text learner, (iii) data instance learner; (iv) context learner. For example, name learner exploits names of elements. It applies tokenization and n-grams to the names in order to create training examples. In addition, the name learner uses edit distance in order to determine similarity between strings of element names. the data instance learner determines whether the values of instances share common patterns. A matcher, called meta-learner, combines the results produced by basic matchers. It uses logistic regression with the help on the stacking technique in order to learn its parameters.

A.3.11 Falcon-AO (2006)

Falcon-AO (Hu et al. [2005], Jian et al. [2005], Qu et al. [2006]) is an automatic tool for aligning OWL ontologies. There are two matchers integrated in Falcon-AO: one is a matcher based on linguistic matching for ontologies (LMO); the other is a matcher based on graph matching for ontologies (GMO). LMO associates with each ontology entity a bag of words which is built from the entity label, the entity annotations as well as the labels of connected entities. The similarity between entities is based on TFIDF. GMO is a bipartite graph matcher. It starts by considering the RDF representation of the ontologies as a bipartite graph which is represented by its adjacent matrix (A and A'). The distance between the ontologies is represented by a distance matrix X and the distance equations between two entities are simply a linear combination of all entities they are adjacent ($X^{t+1} = AX^tA'^T + A^TX^tA'$). GMO takes the alignments generated by LMO as external input and outputs additional alignments. First, LMO is used for assessing the similarity between ontology entities on the basis of their name and text annotation. If the result has a high confidence, then it is directly returned for extracting an alignment. Otherwise, the result is used as input for the GMO matcher, with tries to find an alignment on the basis of the relationships between entities.

A.3.12 SILAS (2007)

SILAS (Simple Instance-based Library-thesaurus Alignment System) (Ossewaarde [2007]) is an instance-based ontology matching, which measures the similarity between subsets annotated with words from different ontologies to match up the concepts described by these words. It identifies overlapping subsets and computes a simple metric to predict semantic relatedness between the concepts of which the subsets are extensions. The procedure of alignment is as follows: (i) the words describing the ontological concepts in the two ontologies were translated into two sets search terms T; (ii) using automated spiders, for each $t_i \in T$, all records tagged with term t_i are retrieved into subset $B_i \in B$; (iii) each subset corresponding to a concept from source ontology is then compared to all subsets corresponding to concepts from target ontology, and each comparison yields an overlap score for each pair B_i, B_j, where B_i is a subset of the source ontology and B_j is a subset according to target ontology. If the overlap between the two sets is sufficiently high so that the confidence score is greater than 0, B_j is considered an alignment candidate for B_i. The overlap is determined by finding the intersection $B_i \setminus B_j$; and (iv) for each B_i, the alignment candidates were scored according to the confidence measurement system, i.e., if the confidence exceeds certain thresholds, the concepts of which subsets B_i and B_j are extensions, are judged related or equivalent. Moreover, a lexical booster is used for ranking alignment candidates: if there is a lower-ranked alignment candidate with a similar name, it gets promoted to the top ranking.

A.3.13 ASMOV (2007)

ASMOV (Automated Semantic Mapping of Ontologies with Validation) (Jean-Mary and Kabuka [2007a], Jean-Mary and Kabuka [2007b]) is an algorithm which iteratively calculates the similarity between concepts for a pair of ontologies by analyzing four features: textual description (id, label, and comment), external structure (parents and children), internal structure (property restrictions for classes; types, domains, and ranges for properties), and individual similarity. The measures obtained by comparing these four features are combined into a single confidence value using a weighted sum. The initial weights were chosen arbitrarily and they are automatically adjusted based on the information contained in the ontologies (in an automated pre-processing phase). For example, when analyzing the textual information in the pre-processing phase, if ASMOV cannot find meaningful words, it decreases the textual similarity weight based on predetermined rules. The iterative alignments are validated by a number of rules and a mapping validation process. Optionally, it is accepted feedback from a user.

A.3.14 Lily (2007)

Lily (Wang and Xu [2007]) is a matching system that exploits linguistic and structural information in semantic graphs of the entities to generate initial alignments. Based on these initial alignments, a subsequent similarity propagation strategy could produce more alignments which often can not be obtained by the previous process. The matching process consists of three main steps: (i) extracting semantic subgraph, it tries to use a semantic subgraph to represent the real meaning for a given entity in the ontology; (ii) computing alignment similarity by analyzing the literal and structural information in the semantic subgraphs, it computes the similarity confidences between entities from different ontologies; and (iii) similarity propagation, which aims to find more alignments that can not be found in the previous processes. To compute the alignment similarity (ii), Lily uses two kinds of descriptions to interpret the concepts and properties: (a) basic description, which is a document consisting of the identifier, label and comments; and (b) semantic description of a concept, which contains the information about class hierarchies, related properties and instances, and semantic description of a property, which contains the information about hierarchies, domains, ranges, restrictions and related instances. For both descriptions, it calculates the similarities of the corresponding parts and all separate similarities are combined with the experiential weights.

A.3.15 SAMBO (2007)

SAMBO (System for Aligning and Merging Biomedical Ontologies) (Lambrix and Tan [2006], Tan and Lambrix [2007]) is a matching tool initially developed to deal with biological ontologies. The algorithm includes several matchers and alignment suggestions are then determined by combining and filtering the results generated by the matchers. The suggestions are then presented to the user who accepts or rejects them. Specifically, SAMBO contains five basic matchers: two terminological matchers, a structure-based matcher, a matcher based on domain knowledge, and a learning matcher. The basic terminological matcher, *Term* contains matching algorithms based on the textual descriptions (names and synonyms) of concepts and relations. In the current implementation, the matcher includes two approximate string matching algorithms, n-gram and edit distance, and a linguistic algorithm (i.e., by comparing the lists of words of which the terms are composed). *Term* computes similarity values by combining the results from these three algorithms using a weighted sum. The matcher *TermWN* is based on *Term*, but uses a general thesaurus, WordNet to enhance the similarity measure by looking up the hypernym relationships of the pairs of words in WordNet. The structural matcher is based on the is-a and part-of hierarchies of the ontologies, i.e., if two concepts lie in similar positions with respect to is-a or part-of hierarchies relative to already aligned concepts in the two ontologies, then they are likely to be similar as well. The knowledge-based matcher is based on the uses of a thesaurus and the similarity of two terms

is determined by their relationship in such thesaurus. The learning matcher computes the similarity between concepts based on the probability that documents (viewed as instances) about one concept are also about the other concept and vice versa. Finally, a weighted sum of the similarity values computed by different matchers represents the final result.

A.3.16 SEMA (2007)

SEMA (Spiliopoulos et al. [2007]) is an iterative matching algorithm which combines lexical, semantic and structural methods. It combines six matching methods, executed in a predefined sequence. Each method in sequence exploits the results of the previous methods, aiming to find additional mapping element pairs. Firstly, the lexical matcher is applied to compute the similarity between local name, label or comment of an OWL class or property. It uses a lexical similarity approach based on clusters of strings and synonymous provided by the WordNet. The second matching method applied is the semantic matching, which aims at discovering and exploiting latent features that reveal the intended meaning of ontology elements. The third method is a standard Vector Space Model based technique where ontology elements are represented as vectors of weights. Each weight corresponds to a word and is being calculated using the TF/IDF measure. The similarity between two vectors is being computed by means of the cosine similarity measure. The fourth mapping method of SEMA is a lexical matching method exploiting the instances of classes. Specifically, two classes are considered to match if the percentage of their mapped instances is above a predefined threshold. The fifth method of SEMA is a structural matching method, which utilizes the mappings produced by the above described matching methods. According to this method, if two classes have at least a pair of matched super classes and a pair of matched sub class, then they are also considered to match. Finally, the property based matcher utilizes the properties' mappings produced by the other methods in order to locate new matching pairs of classes. Specifically, two classes are considered to match, if the percentage of their mapped properties is above a predefined threshold. Thus, the aggregation of the mappings produced by the individual methods is performed through their iterative execution.

Appendix B

Extended Tables

Table B.1: Individual matcher results - proposed matchers.

	Syntactic									Semantic									Structural			Attribute		
	C			A			All			C			A			All			C			A		
Test	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F
101	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.97	0.98	0.91	0.91	0.91	0.94	0.93	0.93	1.0	1.0	1.0	0.03	1.0	0.06
103	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.97	0.98	0.91	0.91	0.91	0.94	0.93	0.93	1.0	1.0	1.0	0.03	0.98	0.06
104	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.97	0.98	0.91	0.91	0.91	0.94	0.93	0.93	1.0	1.0	1.0	0.03	0.98	0.06
201		0.0		1.0	0.02	0.03	1.0	0.01	0.02		0.0		1.0	0.02	0.03	1.0	0.01	0.02		0.0		0.03	1.0	0.06
202		0.0		1.0	0.02	0.03	1.0	0.01	0.02		0.0		1.0	0.02	0.03	1.0	0.01	0.02		0.0		0.03	1.0	0.06
203	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.97	0.98	0.91	0.91	0.91	0.94	0.93	0.93	1.0	1.0	1.0	0.03	1.0	0.06
204	1.0	0.48	0.65	1.0	0.75	0.86	1.0	0.66	0.80	1.0	0.45	0.63	0.88	0.72	0.79	0.91	0.63	0.74	1.0	0.33	0.5	0.03	1.0	0.06
205	1.0	0.18	0.31	0.92	0.17	0.29	0.94	0.18	0.30	0.88	0.21	0.34	0.55	0.17	0.26	0.64	0.19	0.29	1.0	0.09	0.17	0.03	1.0	0.06
206	1.0	0.21	0.35	1.0	0.36	0.53	1.0	0.31	0.47	1.0	0.09	0.17	1.0	0.22	0.36	1.0	0.18	0.30	1.0	0.12	0.22	0.03	1.0	0.06
207	1.0	0.21	0.35	1.0	0.36	0.53	1.0	0.31	0.47	1.0	0.09	0.17	1.0	0.22	0.36	1.0	0.18	0.30	1.0	0.12	0.22	0.03	1.0	0.06
208	1.0	0.48	0.65	1.0	0.75	0.86	1.0	0.66	0.80	1.0	0.45	0.63	0.88	0.72	0.79	0.91	0.63	0.74	1.0	0.33	0.50	0.03	1.0	0.06
209	1.0	0.18	0.31	0.92	0.17	0.29	0.94	0.18	0.30	1.0	0.21	0.35	0.55	0.17	0.26	0.67	0.19	0.29	1.0	0.09	0.17	0.03	1.0	0.06
210	1.0	0.21	0.35	1.0	0.35	0.53	1.0	0.31	0.47	1.0	0.09	0.17	1.0	0.22	0.36	1.0	0.18	0.30	1.0	0.12	0.22	0.03	1.0	0.06
221	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.97	0.98	0.91	0.91	0.91	0.94	0.93	0.93	1.0	0.21	0.35	0.03	1.0	0.06
222	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.97	0.98	0.91	0.91	0.91	0.93	0.92	0.92	1.0	0.52	0.68	0.03	1.0	0.06
223	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.97	0.98	0.89	0.91	0.90	0.93	0.93	0.93	1.0	0.64	0.78	0.03	1.0	0.06
224	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.97	0.98	0.91	0.91	0.91	0.94	0.93	0.93	1.0	1.0	1.0	0.03	1.0	0.06
225	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.97	0.98	0.91	0.91	0.91	0.94	0.93	0.93	1.0	1.0	1.0	0.03	1.0	0.06
228	1.0	1.0	1.0				1.0	1.0	1.0	1.0	0.97	0.98				1.0	0.97	0.98	1.0	1.0	1.0			
230	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.96	0.98	0.87	0.87	0.87	0.92	0.90	0.91	1.0	1.0	1.0	0.02	1.0	0.05
231	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.97	0.98	0.91	0.91	0.91	0.94	0.93	0.93	1.0	1.0	1.0	0.03	1.0	0.06
232	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.97	0.98	0.91	0.91	0.91	0.94	0.93	0.93	1.0	0.21	0.35	0.03	1.0	0.06
233	1.0	1.0	1.0				1.0	1.0	1.0	1.0	0.97	0.98				1.0	0.97	0.98	1.0	0.21	0.35			
236	1.0	1.0	1.0				1.0	1.0	1.0	1.0	0.97	0.98				1.0	0.97	0.98	1.0	1.0	1.0			
237	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.97	0.98	0.91	0.91	0.91	0.93	0.92	0.93	1.0	0.52	0.68	0.03	1.0	0.06
238	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.97	0.98	0.89	0.91	0.90	0.93	0.93	0.93	1.0	0.64	0.78	0.03	1.0	0.06
239	1.0	1.0	1.0	0.0			0.97	1.0	0.98	1.0	0.97	0.98	0.0			0.97	0.97	0.97	1.0	0.52	0.68	0.0		
240	1.0	1.0	1.0	0.0			0.97	1.0	0.99	1.0	0.97	0.98	0.0			0.97	0.97	0.97	1.0	0.64	0.78	0.0		
241	1.0	1.0	1.0				1.0	1.0	1.0	1.0	0.97	0.98				1.0	0.97	0.98	1.0	0.21	0.35			
246	1.0	1.0	1.0	0.0			0.97	1.0	0.98	1.0	0.97	0.98	0.0			0.97	0.97	0.97	1.0	0.52	0.68	0.0		
247	1.0	1.0	1.0	0.0			0.97	1.0	0.99	1.0	0.97	0.98	0.0			0.97	0.97	0.97	1.0	0.64	0.78	0.0		
248		0.0		1.0	0.02	0.03	1.0	0.01	0.02		0.0		1.0	0.02	0.03	1.0	0.01	0.02		0.0		0.03	1.0	0.06
249		0.0		1.0	0.02	0.03	1.0	0.01	0.02	-	0.0	-	1.0	0.02	0.03	1.0	0.01	0.02		0.0	-	0.03	1.0	0.06
250		0.0						0.0			0.0						0.0			0.0				
251		0.0		1.0	0.02	0.03	1.0	0.01	0.02		0.0		1.0	0.02	0.03	1.0	0.01	0.02		0.0		0.03	1.0	0.06
252		0.0		1.0	0.02	0.03	1.0	0.01	0.02		0.0		0.5	0.02	0.03	0.5	0.01	0.02		0.0		0.03	1.0	0.06

Table B.1: Individual matcher results - proposed matchers.

	Syntactic									Semantic									Structural			Attribute		
	C			A			All			C			A			All			C			A		
Test	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F
253		0.0		1.0	0.02	0.03	1.0	0.01	0.02		0.0		1.0	0.02	0.03	1.0	0.01	0.02		0.0		0.03	1.0	0.06
254		0.0						0.0			0.0						0.0			0.0				
257		0.0						0.0			0.0						0.0			0.0				
258		0.0		1.0	0.02	0.03	1.0	0.01	0.02		0.0		1.0	0.02	0.03	1.0	0.01	0.02		0.0		0.03	1.0	0.06
259		0.0		1.0	0.02	0.03	1.0	0.01	0.02		0.0		0.5	0.02	0.03	0.5	0.01	0.02		0.0		0.03	1.0	0.06
260		0.0		0.0			0.0	0.0			0.0		0.0			0.0	0.0			0.0		0.0		
261		0.0		0.0			0.0	0.0			0.0		0.0			0.0	0.0			0.0		0.0		
262		0.0						0.0			0.0						0.0			0.0				
265		0.0		0.0			0.0	0.0			0.0		0.0			0.0	0.0			0.0		0.0		
266		0.0		0.0			0.0	0.0			0.0		0.0			0.0	0.0			0.0		0.0		
301	0.93	0.59	0.72	1.0	0.03	0.05	0.93	0.23	0.37	0.92	0.55	0.69	1.0	0.03	0.05	0.93	0.22	0.35		0.0		0.02	0.63	0.03
302	1.0	0.43	0.61	0.92	0.46	0.61	0.95	0.45	0.61	1.0	0.39	0.56	0.77	0.42	0.54	0.86	0.40	0.55		0.0		0.01	0.54	0.02
303	1.0	0.76	0.87	0.83	0.31	0.45	0.92	0.47	0.62	1.0	0.71	0.83	0.71	0.31	0.43	0.85	0.45	0.59		0.0		0.0	0.44	0.02
304	0.97	0.93	0.95	0.95	0.46	0.62	0.96	0.64	0.77	0.96	0.87	0.91	0.87	0.43	0.58	0.92	0.61	0.73	1.0	0.4	0.57	0.02	0.56	0.04
	1.0	0.53	0.85	0.80	0.54	0.59	0.90	0.53	0.67	0.99	0.51	0.82	0.71	0.48	0.52	0.84	0.49	0.62	1.0	0.34	0.66	0.02	0.95	0.05

Table B.2: Different values of strength for the negative mappings – proposed matchers.

	0.5										1									
	C			A			All				C			A			All			
Test	P	R	F	P	R	F	P	R	F		P	R	F	P	R	F	P	R	F	
101	1.0	1.0	1.0	0.03	1.0	0.06	0.05	1.0	0.09		1.0	0.97	0.98	1.0	0.92	0.96	1.0	0.94	0.97	
103	1.0	1.0	1.0	0.03	1.0	0.06	0.05	1.0	0.09		1.0	0.97	0.98	1.0	0.91	0.95	1.0	0.93	0.96	
104	1.0	1.0	1.0	0.03	1.0	0.06	0.05	1.0	0.09		1.0	0.97	0.98	1.0	0.91	0.95	1.0	0.93	0.96	
201		0.0		0.03	1.0	0.06	0.03	0.66	0.06			0.0		1.0	0.02	0.03	1.0	0.01	0.02	
202		0.0		0.03	1.0	0.06	0.03	0.66	0.06			0.0		1.0	0.02	0.03	1.0	0.01	0.02	
203	1.0	1.0	1.0	0.03	1.0	0.06	0.05	1.0	0.09		1.0	0.97	0.98	1.0	0.92	0.96	1.0	0.94	0.97	
204	1.0	0.48	0.65	0.03	1.0	0.06	0.04	0.82	0.07		1.0	0.30	0.47	1.0	0.72	0.84	1.0	0.58	0.73	
205	1.0	0.24	0.39	0.03	0.84	0.05	0.03	0.64	0.06		1.0	0.06	0.11	0.92	0.17	0.29	0.93	0.13	0.23	
206	1.0	0.21	0.35	0.02	0.95	0.05	0.03	0.70	0.06		1.0	0.06	0.11	1.0	0.20	0.34	1.0	0.15	0.27	
207	1.0	0.21	0.35	0.03	0.95	0.06	0.03	0.70	0.06		1.0	0.06	0.11	1.0	0.20	0.34	1.0	0.15	0.27	
208	1.0	0.48	0.65	0.03	1.0	0.06	0.04	0.82	0.07		1.0	0.30	0.47	1.0	0.72	0.84	1.0	0.58	0.73	
209	1.0	0.24	0.39	0.02	0.84	0.05	0.03	0.64	0.06		1.0	0.06	0.11	0.92	0.17	0.29	0.93	0.13	0.23	
210	1.0	0.21	0.35	0.03	0.95	0.06	0.03	0.70	0.06		1.0	0.06	0.11	1.0	0.20	0.34	1.0	0.15	0.27	
221	1.0	1.0	1.0	0.03	1.0	0.06	0.05	1.0	0.09		1.0	0.21	0.35	1.0	0.92	0.96	1.0	0.68	0.81	
222	1.0	1.0	1.0	0.03	1.0	0.06	0.04	1.0	0.08		1.0	0.48	0.65	1.0	0.92	0.96	1.0	0.78	0.88	
223	1.0	1.0	1.0	0.03	1.0	0.06	0.05	1.0	0.09		1.0	0.64	0.78	1.0	0.92	0.96	1.0	0.82	0.90	
224	1.0	1.0	1.0	0.03	1.0	0.06	0.05	1.0	0.09		1.0	0.97	0.98	1.0	0.92	0.96	1.0	0.94	0.97	
225	1.0	1.0	1.0	0.03	1.0	0.06	0.05	1.0	0.09		1.0	0.97	0.98	1.0	0.92	0.96	1.0	0.94	0.97	
228	1.0	1.0	1.0				1.0	1.0	1.0		1.0	0.97	0.98				1.0	0.97	0.98	
230	1.0	1.0	1.0	0.03	1.0	0.05	0.04	1.0	0.07		1.0	0.96	0.98	1.0	0.89	0.94	1.0	0.92	0.96	
231	1.0	1.0	1.0	0.03	1.0	0.06	0.05	1.0	0.09		1.0	0.97	0.98	1.0	0.92	0.96	1.0	0.94	0.97	
232	1.0	1.0	1.0	0.03	1.0	0.06	0.05	1.0	0.09		1.0	0.21	0.35	1.0	0.92	0.96	1.0	0.68	0.81	
233	1.0	1.0	1.0				1.0	1.0	1.0		1.0	0.21	0.35				1.0	0.21	0.35	
236	1.0	1.0	1.0				1.0	1.0	1.0		1.0	0.97	0.98				1.0	0.97	0.98	
237	1.0	1.0	1.0	0.03	1.0	0.06	0.04	1.0	0.08		1.0	0.48	0.65	1.0	0.92	0.95	1.0	0.78	0.87	
238	1.0	1.0	1.0	0.03	1.0	0.06	0.05	1.0	0.09		1.0	0.63	0.77	1.0	0.92	0.95	1.0	0.82	0.90	
239	1.0	1.0	1.0	0.0			0.45	1.0	0.62		1.0	0.48	0.65	0.0			0.93	0.48	0.64	
240	1.0	1.0	1.0	0.0			0.48	1.0	0.65		1.0	0.64	0.78	0.0			0.95	0.64	0.76	
241	1.0	1.0	1.0				1.0	1.0	1.0		1.0	0.21	0.35				1.0	0.21	0.35	
246	1.0	1.0	1.0	0.0			0.45	1.0	0.62		1.0	0.48	0.65	0.0			0.93	0.48	0.64	
247	1.0	1.0	1.0	0.0			0.48	1.0	0.65		1.0	0.64	0.78	0.0			0.95	0.64	0.76	
248		0.0		0.03	1.0	0.05	0.03	0.65	0.05			0.0		1.0	0.02	0.03	1.0	0.01	0.02	
249		0.0		0.03	1.0	0.05	0.03	0.65	0.056		2	0.0		1.0	0.02	0.03	1.0	0.01	0.02	
250		0.0						0.0				0.0						0.0		
251		0.0		0.03	1.0	0.05	0.03	0.68	0.05			0.0		1.0	0.02	0.03	1.0	0.01	0.02	
252		0.0		0.03	1.0	0.06	0.03	0.66	0.06			0.0		1.0	0.02	0.03	1.0	0.01	0.02	

Table B.2: Different values of strength for the negative mappings – proposed matchers.

	0.5									1								
	C			A			All			C			A			All		
Test	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F
253		0.0		0.03	1.0	0.06	0.03	0.66	0.06		0.0		1.0	0.02	0.03	1.0	0.01	0.02
254		0.0						0.0			0.0						0.0	
257		0.0						0.0			0.0						0.0	
258		0.0		0.03	1.0	0.06	0.03	0.69	0.06		0.0		1.0	0.02	0.03	1.0	0.01	0.02
259		0.0		0.03	1.0	0.06	0.03	0.66	0.06		0.0		1.0	0.02	0.03	1.0	0.01	0.02
260		0.0		0.0			0.0	0.0			0.0		0.0			0.0	0.0	
261		0.0		0.0			0.0	0.0			0.0		0.0			0.0	0.0	
262		0.0						0.0			0.0						0.0	
265		0.0		0.0			0.0	0.0			0.0		0.0			0.0	0.0	
266		0.0		0.0			0.0	0.0			0.0		0.0			0.0	0.0	
301	0.93	0.59	0.72	0.01	0.21	0.01	0.01	0.35	0.03		0.0		1.0	0.03	0.05	1.0	0.02	0.03
302	1.0	0.43	0.61	0.01	0.50	0.02	0.02	0.47	0.04				0.91	0.42	0.57	0.91	0.21	0.34
303	1.0	0.76	0.87	0.01	0.44	0.02	0.02	0.55	0.03		0.0		0.83	0.31	0.45	0.83	0.20	0.33
304	0.97	0.93	0.95	0.02	0.50	0.03	0.04	0.67	0.07	1.0	0.37	0.54	0.95	0.46	0.62	0.97	0.42	0.59
H-mean	1.0	0.54	0.86	0.02	0.92	0.05	0.15	0.70	0.21	0.99	0.33	0.63	0.80	0.48	0.54	0.90	0.39	0.53

Table B.3: Baseline, negotiation, and argumentation results.

	Baseline									Negotiation									Argumentation								
	C			A			All			C			A			All			C			A			All		
Test	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F
101	1.0	1.0	1.0	0.03	1.0	0.06	0.05	1.0	0.09	1.0	1.0	1.0	0.91	1.0	0.96	0.94	1.0	0.97	1.0	0.96	0.98	1.0	0.92	0.95	1.0	0.93	0.96
103	1.0	1.0	1.0	0.03	1.0	0.06	0.05	1.0	0.09	1.0	1.0	1.0	0.91	1.0	0.96	0.94	1.0	0.97	1.0	0.96	0.98	1.0	0.90	0.95	1.0	0.92	0.96
104	1.0	1.0	1.0	0.03	1.0	0.06	0.05	1.0	0.09	1.0	1.0	1.0	0.91	1.0	0.96	0.94	1.0	0.97	1.0	0.96	0.98	1.0	0.90	0.95	1.0	0.92	0.96
201		0.0		0.03	1.0	0.06	0.03	0.66	0.06		0.0		1.0	0.02	0.03	1.0	0.01	0.02		0.0	-	1.0	0.01	0.03	1.0	0.01	0.02
202		0.0		0.03	1.0	0.06	0.03	0.66	0.06		0.0		1.0	0.02	0.03	1.0	0.01	0.02	-	0.0	-	1.0	0.01	0.03	1.0	0.01	0.02
203	1.0	1.0	1.0	0.03	1.0	0.06	0.05	1.0	0.09	1.0	1.0	1.0	0.91	1.0	0.96	0.94	1.0	0.97	1.0	0.96	0.98	1.0	0.92	0.95	1.0	0.93	0.96
204	1.0	0.48	0.65	0.03	1.0	0.06	0.04	0.82	0.07	1.0	0.48	0.65	0.89	0.75	0.81	0.91	0.66	0.77	1.0	0.30	0.46	1.0	0.71	0.83	1.0	0.57	0.73
205	0.89	0.24	0.38	0.03	0.84	0.05	0.03	0.64	0.06	1.0	0.18	0.31	0.55	0.17	0.26	0.65	0.18	0.28	1.0	0.06	0.11	0.91	0.17	0.28	0.92	0.13	0.23
206	1.0	0.21	0.35	0.03	0.95	0.06	0.03	0.70	0.06	1.0	0.15	0.26	1.0	0.38	0.55	1.0	0.30	0.46	1.0	0.06	0.11	1.0	0.205	0.33	1.0	0.156	0.26
207	1.0	0.21	0.35	0.03	0.95	0.06	0.03	0.70	0.06	1.0	0.15	0.26	1.0	0.38	0.55	1.0	0.30	0.46	1.0	0.06	0.11	1.0	0.20	0.33	1.0	0.15	0.26
208	1.0	0.48	0.65	0.03	1.0	0.06	0.04	0.82	0.07	1.0	0.48	0.65	0.89	0.75	0.81	0.91	0.66	0.77	1.0	0.30	0.46	1.0	0.71	0.83	1.0	0.57	0.73
209	1.0	0.24	0.39	0.03	0.84	0.05	0.03	0.64	0.06	1.0	0.18	0.31	0.55	0.17	0.26	0.65	0.18	0.28	1.0	0.06	0.11	0.91	0.17	0.28	0.92	0.13	0.23
210	1.0	0.21	0.35	0.02	0.95	0.06	0.03	0.70	0.06	1.0	0.15	0.26	1.0	0.38	0.55	1.0	0.30	0.46	1.0	0.06	0.11	1.0	0.20	0.33	1.0	0.15	0.26
221	1.0	1.0	1.0	0.03	1.0	0.06	0.05	1.0	0.09	1.0	0.97	0.98	0.91	1.0	0.96	0.94	0.98	0.97	1.0	0.21	0.35	1.0	0.92	0.95	1.0	0.68	0.80
222	1.0	1.0	1.0	0.03	1.0	0.06	0.04	1.0	0.08	1.0	1.0	1.0	0.91	1.0	0.96	0.94	1.0	0.97	1.0	0.48	0.65	1.0	0.92	0.95	1.0	0.78	0.87
223	1.0	1.0	1.0	0.03	1.0	0.06	0.05	1.0	0.09	1.0	0.97	0.98	0.90	1.0	0.95	0.93	0.99	0.96	1.0	0.63	0.77	1.0	0.92	0.95	1.0	0.82	0.90
224	1.0	1.0	1.0	0.03	1.0	0.06	0.05	1.0	0.09	1.0	1.0	1.0	0.91	1.0	0.96	0.94	1.0	0.97	1.0	0.96	0.98	1.0	0.92	0.95	1.0	0.93	0.96
225	1.0	1.0	1.0	0.03	1.0	0.06	0.05	1.0	0.09	1.0	1.0	1.0	0.91	1.0	0.96	0.94	1.0	0.97	1.0	0.96	0.98	1.0	0.92	0.95	1.0	0.93	0.96
228	1.0	1.0	1.0				1.0	1.0	1.0	1.0	1.0	1.0				1.0	1.0	1.0	1.0	0.96	0.98	-	-	-	1.0	0.96	0.98
230	1.0	1.0	1.0	0.03	1.0	0.05	0.04	1.0	0.07	1.0	1.0	1.0	0.89	1.0	0.94	0.92	1.0	0.96	1.0	0.96	0.97	1.0	0.89	0.94	1.0	0.91	0.95
231	1.0	1.0	1.0	0.03	1.0	0.06	0.05	1.0	0.09	1.0	1.0	1.0	0.91	1.0	0.96	0.94	1.0	0.97	1.0	0.96	0.98	1.0	0.92	0.95	1.0	0.93	0.96
232	1.0	1.0	1.0	0.03	1.0	0.06	0.05	1.0	0.09	1.0	0.97	0.98	0.91	1.0	0.96	0.94	0.99	0.96	1.0	0.21	0.35	1.0	0.92	0.95	1.0	0.68	0.80
233	1.0	1.0	1.0				1.0	1.0	1.0	1.0	0.97	0.98				1.0	0.97	0.98	1.0	0.21	0.35	-	-	-	1.0	0.21	0.35
236	1.0	1.0	1.0				1.0	1.0	1.0	1.0	1.0	1.0				1.0	1.0	1.0	1.0	0.96	0.98	-	-	-	1.0	0.96	0.98
237	1.0	1.0	1.0	0.03	1.0	0.06	0.04	1.0	0.08	1.0	1.0	1.0	0.91	1.0	0.96	0.94	1.0	0.97	1.0	0.48	0.65	1.0	0.92	0.95	1.0	0.78	0.87
238	1.0	1.0	1.0	0.03	1.0	0.06	0.05	1.0	0.09	1.0	0.97	0.98	0.90	1.0	0.95	0.93	0.99	0.96	1.0	0.63	0.77	1.0	0.92	0.95	1.0	0.82	0.90
239	1.0	1.0	1.0	0.0	-	-	0.45	1.0	0.62	1.0	1.0	1.0	0.0			0.97	1.0	0.98	1.0	0.48	0.65	0.0	-	-	0.93	0.48	0.63
240	1.0	1.0	1.0	0.0	-	-	0.48	1.0	0.65	1.0	0.97	0.98	0.0	-	-	0.97	0.97	0.97	1.0	0.63	0.77	0.0	-	-	0.95	0.63	0.76
241	1.0	1.0	1.0				1.0	1.0	1.0	1.0	0.97	0.98				1.0	0.97	0.98	1.0	0.21	0.35	-	-	-	1.0	0.21	0.35
246	1.0	1.0	1.0	0.0			0.45	1.0	0.62	1.0	1.0	1.0	0.0			0.97	1.0	0.98	1.0	0.48	0.65	0.0	-	-	0.93	0.48	0.63
247	1.0	1.0	1.0	0.0			0.48	1.0	0.65	1.0	0.97	0.98	0.0			0.97	0.97	0.97	1.0	0.63	0.77	0.0	-	-	0.95	0.63	0.76
248		0.0		0.03	1.0	0.06	0.03	0.66	0.06		0.0		1.0	0.02	0.03	1.0	0.01	0.02	-	0.0	-	1.0	0.01	0.03	1.0	0.01	0.02
249		0.0		0.03	1.0	0.06	0.03	0.66	0.06		0.0		1.0	0.02	0.03	1.0	0.01	0.02	-	0.0	-	1.0	0.01	0.03	1.0	0.01	0.02
250		0.0						0.0			0.0						0.0		-	0.0	-	-	-	-	-	0.0	-
251		0.0		0.03	1.0	0.06	0.03	0.69	0.06		0.0		1.0	0.02	0.03	1.0	0.01	0.02	-	0.0	-	1.0	0.01	0.03	1.0	0.01	0.02
252		0.0		0.03	1.0	0.06	0.03	0.66	0.06		0.0		0.5	0.02	0.03	0.5	0.01	0.02	-	0.0	-	1.0	0.01	0.03	1.0	0.01	0.02
253		0.0		0.03	1.0	0.06	0.03	0.66	0.06		0.0		1.0	0.02	0.03	1.0	0.01	0.02	-	0.0	-	1.0	0.01	0.03	1.0	0.01	0.02

Table B.3: Baseline, negotiation, and argumentation results - proposed matchers.

	Baseline									Negotiation									Argumentation								
	C			A			All			C			A			All			C			A			All		
Test	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F
254		0.0						0.0			0.0						0.0		-	0.0	-	-	-	-	-	0.0	-
257		0.0						0.0			0.0						0.0		-	0.0	-	-	-	-	-	0.0	-
258		0.0		0.03	1.0	0.06	0.03	0.69	0.06		0.0		1.0	0.02	0.03	1.0	0.01	0.02	-	0.0	-	1.0	0.01	0.03	1.0	0.01	0.02
259		0.0		0.03	1.0	0.06	0.03	0.66	0.06		0.0		0.5	0.02	0.03	0.5	0.01	0.02	-	0.0	-	1.0	0.01	0.03	1.0	0.01	0.02
260		0.0		0.0			0.0	0.0			0.0		0.0			0.0	0.0		-	0.0	-	0.0	-	-	0.0	0.0	-
261		0.0		0.0			0.0	0.0			0.0		0.0			0.0	0.0	-	-	0.0	-	0.0	-	-	0.0	0.0	-
262		0.0						0.0			0.0						0.0		-	0.0	-	-	-	-	-	0.0	-
265		0.0		0.0			0.0	0.0			0.0		0.0			0.0	0.0		-	0.0	-	0.0	-	-	0.0	0.0	-
266		0.0		0.0			0.0	0.0			0.0		0.0			0.0	0.0		-	0.0	-	0.0	-	-	0.0	0.0	-
301	0.93	0.59	0.72	0.01	0.21	0.01	0.01	0.35	0.03	0.92	0.55	0.69	1.0	0.03	0.05	0.93	0.22	0.35	-	0.0	-	1.0	0.02	0.05	1.0	0.01	0.03
302	1.0	0.43	0.61	0.01	0.5	0.02	0.02	0.47	0.04	1.0	0.39	0.56	0.79	0.46	0.58	0.87	0.43	0.57	-	0.0	-	0.90	0.41	0.57	0.90	0.21	0.34
303	1.0	0.76	0.87	0.01	0.44	0.02	0.02	0.55	0.03	1.0	0.71	0.83	0.71	0.31	0.43	0.85	0.45	0.59	-	0.0	-	0.83	0.31	0.45	0.83	0.20	0.32
304	0.97	0.93	0.95	0.02	0.50	0.03	0.04	0.67	0.07	0.96	0.90	0.93	0.88	0.46	0.60	0.92	0.63	0.75	1.0	0.36	0.53	0.95	0.45	0.61	0.96	0.42	0.58
	0.99	0.54	0.86	0.02	0.92	0.05	0.15	0.70	0.21	1.0	0.52	0.84	0.71	0.54	0.56	0.84	0.52	0.65	1.0	0.32	0.63	0.80	0.48	0.54	0.90	0.39	0.53

Table B.4: Individual results - OAEI matchers (I).

	ASMOV			DSSim			Falcon			Lily			Ola			OntoDNA		
Test	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F
101	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.94	1.0	0.97
103	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.94	1.0	0.97
104	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.94	1.0	0.97
201	1.0	1.0	1.0	1.0	0.16	0.28	1.0	0.95	0.97	1.0	1.0	1.0	0.85	0.85	0.85	0.11	0.01	0.02
202	0.88	0.88	0.88	1.0	0.16	0.28	0.87	0.87	0.87	1.0	0.80	0.89	0.84	0.84	0.84	0.11	0.01	0.02
203	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.94	1.0	0.97
204	1.0	1.0	1.0	0.96	0.91	0.93	0.98	0.98	0.98	1.0	1.0	1.0	1.0	1.0	1.0	0.93	0.84	0.88
205	1.0	1.0	1.0	0.94	0.33	0.49	1.0	0.98	0.99	1.0	0.99	0.99	0.92	0.92	0.92	0.57	0.12	0.20
206	1.0	0.99	0.99	0.97	0.39	0.56	1.0	0.93	0.96	1.0	0.99	0.99	0.99	0.99	0.99	0.69	0.23	0.34
207	1.0	0.99	0.99	0.97	0.39	0.56	0.98	0.91	0.94	1.0	0.99	0.99	0.97	0.97	0.97	0.69	0.23	0.34
208	1.0	1.0	1.0	0.95	0.90	0.92	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.93	0.84	0.88
209	0.92	0.90	0.91	0.91	0.32	0.47	0.79	0.78	0.79	0.92	0.91	0.91	0.70	0.69	0.69	0.57	0.12	0.20
210	0.97	0.95	0.96	0.97	0.39	0.56	0.81	0.80	0.81	1.0	0.91	0.95	0.87	0.87	0.87	0.69	0.23	0.34
221	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.93	0.76	0.84
222	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.99	1.0	0.99	0.94	1.0	0.97
223	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.99	1.0	0.99	0.94	1.0	0.97
224	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.99	0.99	1.0	1.0	1.0	1.0	1.0	1.0	0.94	1.0	0.97
225	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.94	1.0	0.97
228	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.53	0.27	0.36
230	0.99	1.0	0.99	0.97	1.0	0.99	0.94	1.0	0.97	0.94	1.0	0.97	0.92	1.0	0.96	0.91	1.0	0.95
231	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.94	1.0	0.97
232	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.99	0.99	1.0	1.0	1.0	0.99	1.0	0.99	0.93	0.76	0.84
233	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.53	0.27	0.36
236	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.53	0.27	0.36
237	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.99	0.99	1.0	1.0	1.0	0.98	1.0	0.99	0.94	1.0	0.97
238	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.99	0.99	0.98	0.98	0.98	1.0	1.0	1.0	0.94	1.0	0.97
239	0.97	1.0	0.98	0.97	1.0	0.98	1.0	1.0	1.0	0.97	1.0	0.98	0.94	1.0	0.97	0.50	0.31	0.38
240	0.97	1.0	0.99	0.97	1.0	0.99	1.0	1.0	1.0	0.97	1.0	0.99	0.94	1.0	0.97	0.50	0.27	0.35
241	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.53	0.27	0.36
246	0.97	1.0	0.98	0.97	1.0	0.98	1.0	1.0	1.0	0.97	1.0	0.98	0.94	1.0	0.97	0.50	0.31	0.38
247	0.94	0.97	0.96	0.97	1.0	0.99	1.0	1.0	1.0	0.94	0.97	0.96	0.94	1.0	0.97	0.50	0.27	0.35
248	0.86	0.82	0.84	1.0	0.16	0.28	0.85	0.84	0.84	1.0	0.77	0.87	0.84	0.78	0.81	0.11	0.01	0.02
249	0.89	0.89	0.89	1.0	0.16	0.28	0.87	0.87	0.87	1.0	0.80	0.89	0.79	0.79	0.79	0.11	0.01	0.02
250	0.91	0.30	0.45	1.0	0.27	0.43	1.0	0.27	0.43	0.85	0.67	0.75	0.75	0.27	0.40	0.0	0.0	
251	0.83	0.78	0.80	1.0	0.17	0.29	0.56	0.56	0.56	0.96	0.74	0.84	0.73	0.74	0.74	0.11	0.01	0.02
252	0.87	0.87	0.87	1.0	0.16	0.28	0.71	0.71	0.71	0.94	0.76	0.84	0.79	0.79	0.79	0.11	0.01	0.02
253	0.85	0.81	0.83	1.0	0.16	0.28	0.85	0.84	0.84	0.97	0.75	0.85	0.79	0.74	0.77	0.11	0.01	0.02
254	0.83	0.30	0.44	1.0	0.27	0.43	1.0	0.27	0.43	1.0	0.27	0.43	0.9	0.27	0.42	0.0	0.0	

Table B.4: Individual results - OAEI matchers (I).

	ASMOV			DSSim			Falcon			Lily			Ola			OntoDNA		
Test	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F
257	0.91	0.30	0.45	1.0	0.27	0.43	1.0	0.27	0.43	0.8	0.67	0.75	0.75	0.27	0.40	0.0	0.0	
258	0.82	0.76	0.79	1.0	0.17	0.29	0.54	0.54	0.54	0.76	0.74	0.75	0.74	0.75	0.75	0.11	0.01	0.02
259	0.87	0.87	0.87	1.0	0.16	0.28	0.70	0.70	0.70	0.94	0.75	0.83	0.80	0.80	0.8	0.11	0.01	0.02
260	0.78	0.24	0.37	0.90	0.31	0.46	1.0	0.31	0.47	0.62	0.45	0.52	0.77	0.34	0.48	0.0	0.0	
261	0.91	0.30	0.45	0.90	0.27	0.42	0.89	0.24	0.38	0.61	0.42	0.5	0.69	0.27	0.39	0.0	0.0	
262	0.83	0.30	0.44	1.0	0.27	0.43	1.0	0.27	0.43	1.0	0.27	0.43	0.90	0.27	0.42	0.0	0.0	
265	0.77	0.34	0.48	0.90	0.31	0.46	1.0	0.31	0.47	0.86	0.41	0.56	0.69	0.31	0.43	0.0	0.0	
266	0.91	0.30	0.45	0.80	0.24	0.37	0.89	0.24	0.38	0.64	0.42	0.51	0.69	0.27	0.39	0.0	0.0	
301	0.91	0.82	0.86	0.82	0.30	0.44	0.89	0.82	0.85	0.87	0.80	0.83	0.70	0.65	0.67	0.85	0.68	0.76
302	0.66	0.57	0.61	0.82	0.60	0.69	0.87	0.57	0.69	0.79	0.64	0.71	0.49	0.49	0.49	0.86	0.38	0.53
303	0.75	0.86	0.80	0.85	0.80	0.82	0.77	0.76	0.76	0.58	0.69	0.63	0.41	0.82	0.54	0.90	0.78	0.84
304	0.95	0.96	0.95	0.95	0.92	0.94	0.96	0.93	0.95	0.91	0.97	0.94	0.89	0.97	0.93	0.92	0.88	0.90
H-mean	0.93	0.84	0.87	0.97	0.64	0.71	0.93	0.81	0.84	0.94	0.85	0.88	0.88	0.81	0.83	0.54	0.42	0.54

Table B.5: Individual results - OAEI matchers (II).

	PriorPlus			RiMOM			Sambo			SEMA			TaxoMap			XSom		
Test	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F
101	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.98	0.99	1.0	1.0	1.0	1.0	0.34	0.51	1.0	0.98	0.99
103	1.0	1.0	1.0	1.0	1.0	1.0	0.99	0.97	0.98	1.0	1.0	1.0	1.0	0.34	0.51	1.0	0.98	0.99
104	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.98	0.99	1.0	1.0	1.0	1.0	0.34	0.51	0.97	1.0	0.98
201	0.96	0.94	0.95	1.0	1.0	1.0	0.92	0.23	0.36	0.92	0.98	0.95	1.0	0.06	0.12	0.81	0.81	0.81
202	0.87	0.63	0.73	1.0	0.80	0.89	1.0	0.01	0.02	0.74	0.30	0.43		0.0		0.82	0.82	0.82
203	1.0	1.0	1.0	1.0	0.88	0.93	1.0	0.84	0.91	1.0	1.0	1.0		0.0			0.0	
204	1.0	1.0	1.0	1.0	1.0	1.0	0.99	0.82	0.90	0.95	0.96	0.95	0.92	0.24	0.38	0.99	0.69	0.81
205	0.97	0.95	0.96	1.0	0.99	0.99	0.95	0.37	0.53	0.93	0.96	0.94	0.77	0.10	0.18	0.72	0.71	0.72
206	0.98	0.96	0.97	1.0	0.99	0.99	1.0	0.42	0.59	0.94	0.97	0.95		0.0		0.74	0.73	0.74
207	0.98	0.96	0.97	1.0	0.99	0.99	1.0	0.42	0.59	0.92	0.97	0.94		0.0		0.69	0.68	0.68
208	1.0	0.95	0.97	0.98	0.86	0.91	1.0	0.71	0.83	0.89	0.80	0.84		0.0		0.99	0.75	0.85
209	0.83	0.59	0.69	1.0	0.84	0.91	0.95	0.22	0.35	0.82	0.61	0.70		0.0		0.70	0.69	0.69
210	0.97	0.79	0.88	0.99	0.85	0.91	1.0	0.23	0.37	0.81	0.47	0.60		0.0		0.70	0.69	0.69
221	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.98	0.99	1.0	1.0	1.0	1.0	0.34	0.51	1.0	0.99	0.99
222	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.98	0.99	0.96	1.0	0.98	1.0	0.31	0.48	1.0	0.98	0.99
223	1.0	1.0	1.0	1.0	1.0	1.0	0.98	0.94	0.96	0.97	0.98	0.97	0.68	0.31	0.43	1.0	0.98	0.99
224	1.0	1.0	1.0	1.0	0.99	0.99	0.99	0.97	0.98	1.0	1.0	1.0	1.0	0.34	0.51	1.0	0.98	0.99
225	1.0	1.0	1.0	1.0	1.0	1.0	0.99	0.97	0.98	1.0	1.0	1.0	1.0	0.34	0.51	1.0	0.98	0.99
228	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
230	0.94	1.0	0.97	0.94	1.0	0.97	0.91	0.96	0.93	0.75	1.0	0.86	1.0	0.35	0.52	0.99	0.97	0.98
231	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.98	0.99	1.0	1.0	1.0	1.0	0.34	0.51		0.0	
232	1.0	1.0	1.0	1.0	0.99	0.99	0.99	0.97	0.98	1.0	1.0	1.0	1.0	0.34	0.51	1.0	0.99	0.99
233	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
236	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.97	1.0	0.99
237	0.99	1.0	0.99	1.0	0.99	0.99	1.0	0.98	0.99	0.96	1.0	0.98	1.0	0.31	0.48	1.0	0.98	0.99
238	1.0	1.0	1.0	1.0	0.99	0.99	0.98	0.94	0.96	0.97	0.97	0.97	0.68	0.31	0.43	1.0	0.99	0.99
239	0.97	1.0	0.98	1.0	1.0	1.0	0.97	1.0	0.98	0.94	1.0	0.97	1.0	1.0	1.0	0.97	1.0	0.98
240	0.97	1.0	0.99	1.0	1.0	1.0	0.94	0.91	0.92	1.0	1.0	1.0	0.68	0.91	0.78	0.97	1.0	0.99
241	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
246	0.97	1.0	0.98	1.0	1.0	1.0	0.97	1.0	0.98	0.94	1.0	0.97	1.0	1.0	1.0	0.97	1.0	0.98
247	0.97	1.0	0.99	1.0	1.0	1.0	0.94	0.91	0.92	0.94	0.94	0.94	0.68	0.91	0.78	0.97	1.0	0.99
248	0.76	0.45	0.57	0.99	0.78	0.87	1.0	0.01	0.02	0.73	0.30	0.42		0.0		0.75	0.75	0.75
249	0.88	0.63	0.73	1.0	0.79	0.89	1.0	0.01	0.02	0.73	0.28	0.40		0.0		0.60	0.60	0.60
250	0.68	0.58	0.62	1.0	0.55	0.71		0.0		1.0	0.27	0.43		0.0		0.18	0.18	0.18
251	0.71	0.40	0.51	0.76	0.58	0.66	1.0	0.01	0.02	0.65	0.26	0.37		0.0		0.45	0.45	0.45
252	0.69	0.39	0.5	0.85	0.70	0.77	1.0	0.01	0.02	0.65	0.25	0.36		0.0		0.49	0.49	0.49
253	0.76	0.45	0.57	0.99	0.77	0.87	1.0	0.01	0.02	0.71	0.28	0.40		0.0		0.54	0.54	0.54
254	1.0	0.27	0.43	1.0	0.27	0.43		0.0		1.0	0.27	0.43		0.0		0.03	0.03	0.03

Table B.5: Individual results - OAEI matchers (II).

	PriorPlus			RiMON			Sambo			SEMA			TaxoMap			XSom		
Test	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F
257	0.69	0.61	0.65	1.0	0.55	0.71		0.0		1.0	0.27	0.43		0.0		0.12	0.12	0.12
258	0.71	0.39	0.50	0.76	0.57	0.65	1.0	0.01	0.02	0.66	0.25	0.36		0.0		0.32	0.32	0.32
259	0.73	0.37	0.49	0.85	0.69	0.76	1.0	0.01	0.02	0.68	0.26	0.37		0.0		0.33	0.33	0.33
260	0.67	0.34	0.45	0.93	0.45	0.60	0.0	0.0		1.0	0.31	0.47		0.0		0.03	0.03	0.03
261	0.45	0.39	0.42	1.0	0.27	0.43	0.0	0.0		1.0	0.27	0.43		0.0		0.03	0.03	0.03
262	1.0	0.27	0.43	1.0	0.27	0.43		0.0		1.0	0.27	0.43		0.0		0.0	0.0	
265	0.67	0.34	0.45	0.93	0.45	0.60	0.0	0.0		1.0	0.31	0.47		0.0		0.0	0.0	
266	0.45	0.39	0.42	1.0	0.27	0.43	0.0	0.0		1.0	0.27	0.43		0.0		0.03	0.03	0.03
301	0.91	0.82	0.86	0.73	0.67	0.70	0.95	0.70	0.81	0.68	0.75	0.71	1.0	0.22	0.36	0.91	0.50	0.65
302	0.79	0.66	0.72	0.70	0.64	0.67	0.90	0.19	0.32	0.60	0.60	0.60	1.0	0.21	0.35	0.96	0.57	0.72
303	0.81	0.80	0.80	0.45	0.86	0.59	0.90	0.76	0.82	0.55	0.80	0.65	0.80	0.24	0.38	0.90	0.73	0.81
304	0.90	0.97	0.94	0.90	0.97	0.94	0.96	0.89	0.93	0.77	0.93	0.85	0.93	0.34	0.50	0.96	0.87	0.91
H-mean	0.89	0.79	0.82	0.95	0.83	0.86	0.89	0.55	0.69	0.89	0.72	0.76	0.93	0.27	0.58	0.72	0.66	0.73

Table B.6: Different values of strength for the negative mappings and *original* strength for positive mappings – OAEI matchers.

	0.5			1		
Test	P	R	F	P	R	F
101	1.0	1.0	1.0	1.0	0.34	0.51
103	0.99	1.0	0.99	1.0	0.34	0.51
104	0.97	1.0	0.98	1.0	0.34	0.51
201	0.71	1.0	0.83		0.0	
202	0.59	0.90	0.71		0.0	
203	1.0	1.0	1.0		0.0	
204	0.92	1.0	0.96	1.0	0.16	0.28
205	0.70	1.0	0.82	1.0	0.01	0.02
206	0.75	1.0	0.86		0.0	
207	0.71	1.0	0.83		0.0	
208	0.89	1.0	0.94		0.0	
209	0.57	0.98	0.72		0.0	
210	0.65	0.95	0.77		0.0	
221	1.0	1.0	1.0	1.0	0.1	0.19
222	0.96	1.0	0.98	1.0	0.31	0.48
223	0.95	1.0	0.97	1.0	0.31	0.47
224	0.99	1.0	0.99	1.0	0.34	0.51
225	0.99	1.0	0.99	1.0	0.34	0.51
228	1.0	1.0	1.0	1.0	0.27	0.43
230	0.73	1.0	0.84	1.0	0.35	0.52
231	1.0	1.0	1.0		0.0	
232	0.99	1.0	0.99	1.0	0.1	0.19
233	1.0	1.0	1.0	1.0	0.27	0.43
236	0.97	1.0	0.99	1.0	0.27	0.43
237	0.96	1.0	0.98	1.0	0.31	0.48
238	0.94	1.0	0.97	1	0.31	0.47
239	0.91	1.0	0.95	1.0	0.31	0.47
240	0.92	1.0	0.96	1.0	0.21	0.35

Table B.6: Different values of strength for the negative mappings and *original* strength for positive mappings – OAEI matchers..

	0.5			1		
Test	P	R	F	P	R	F
241	1.0	1.0	1.0	1.0	0.27	0.43
246	0.91	1.0	0.95	1.0	0.31	0.47
247	0.89	1.0	0.94	1.0	0.21	0.35
248	0.55	0.92	0.68		0.0	
249	0.52	0.90	0.66		0.0	
250	0.39	0.73	0.51		0.0	
251	0.44	0.86	0.58		0.0	
252	0.46	0.89	0.60		0.0	
253	0.46	0.90	0.61		0.0	
254	0.24	0.30	0.27		0.0	
257	0.38	0.70	0.49		0.0	
258	0.38	0.88	0.53		0.0	
259	0.43	0.91	0.58		0.0	
260	0.28	0.55	0.37		0.0	
261	0.25	0.55	0.34		0.0	
262	0.21	0.27	0.24		0.0	
265	0.28	0.48	0.35		0.0	
266	0.25	0.55	0.35		0.0	
301	0.56	0.83	0.67	1.0	0.13	0.24
302	0.43	0.70	0.53		0.0	
303	0.42	0.86	0.56	1.0	0.2	0.34
304	0.72	0.97	0.77	1.0	0.13	0.40
H-mean	0.70	0.89	0.77	1.0	0.13	0.40

Table B.7: Baseline, negotiation, and argumentation results - OAEI matchers.

	Baseline			Negotiation			Argumentation		
Test	P	R	F	P	R	F	P	R	F
101	0.94	1.0	0.97	1.0	1.0	1.0	1.0	1.0	1.0
103	0.93	1.0	0.97	1.0	1.0	1.0	0.99	1.0	0.99
104	0.92	1.0	0.96	1.0	1.0	1.0	0.97	1.0	0.98
201	0.66	1.0	0.80	1.0	0.99	0.99	0.71	1.0	0.83
202	0.53	0.92	0.67	1.0	0.78	0.88	0.59	0.90	0.71
203	0.94	1.0	0.97	1.0	1.0	1.0	1.0	1.0	1.0
204	0.86	1.0	0.92	1.0	1.0	1.0	0.92	1.0	0.96
205	0.63	1.0	0.77	1.0	0.98	0.99	0.70	1.0	0.82
206	0.68	1.0	0.81	1.0	0.98	0.99	0.75	1.0	0.86
207	0.65	1.0	0.79	1.0	0.97	0.98	0.71	1.0	0.83
208	0.82	1.0	0.90	1.0	0.98	0.99	0.89	1.0	0.94
209	0.45	0.98	0.62	1.0	0.75	0.86	0.57	0.98	0.72
210	0.54	0.96	0.69	1.0	0.84	0.91	0.65	0.95	0.77
221	0.94	1.0	0.97	1.0	1.0	1.0	1.0	1.0	1.0
222	0.89	1.0	0.64	1.0	1.0	1.0	0.96	1.0	0.98
223	0.80	1.0	0.89	1.0	1.0	1.0	0.95	1.0	0.97
224	0.93	1.0	0.97	1.0	1.0	1.0	0.99	1.0	0.99
225	0.93	1.0	0.97	1.0	1.0	1.0	0.99	1.0	0.99
228	0.80	1.0	0.89	1.0	1.0	1.0	1.0	1.0	1.0
230	0.67	1.0	0.80	0.95	1.0	0.97	0.73	1.0	0.84
231	0.94	1.0	0.97	1.0	1.0	1.0	1.0	1.0	1.0
232	0.92	1.0	0.96	1.0	1.0	1.0	0.99	1.0	0.99
233	0.80	1.0	0.89	1.0	1.0	1.0	1.0	1.0	1.0
236	0.79	1.0	0.88	1.0	1.0	1.0	0.97	1.0	0.99
237	0.88	1.0	0.93	1.0	1.0	1.0	0.96	1.0	0.98
238	0.80	1.0	0.89	1.0	1.0	1.0	0.94	1.0	0.97
239	0.71	1.0	0.83	0.97	1.0	0.98	0.91	1.0	0.95
240	0.58	1.0	0.73	0.97	1.0	0.98	0.92	1.0	0.96

Table B.7: Baseline, negotiation, and argumentation results - OAEI matchers.

	Baseline			Negotiation			Argumentation		
Test	P	R	F	P	R	F	P	R	F
241	0.80	1.0	0.89	1.0	1.0	1.0	1.0	1.0	1.0
246	0.71	1.0	0.83	0.97	1.0	0.98	0.91	1.0	0.95
247	0.56	1.0	0.72	0.97	1.0	0.98	0.89	1.0	0.94
248	0.49	0.93	0.64	1.0	0.68	0.81	0.55	0.92	0.68
249	0.47	0.91	0.62	1.0	0.72	0.84	0.52	0.90	0.66
250	0.35	0.79	0.48	1.0	0.27	0.43	0.39	0.73	0.51
251	0.33	0.87	0.48	1.0	0.47	0.64	0.44	0.86	0.58
252	0.37	0.89	0.52	1.0	0.57	0.72	0.46	0.89	0.60
253	0.42	0.91	0.57	1.0	0.59	0.74	0.46	0.90	0.61
254	0.20	0.33	0.25	1.0	0.27	0.43	0.24	0.30	0.27
257	0.32	0.70	0.43	1.0	0.27	0.43	0.38	0.70	0.49
258	0.30	0.89	0.45	1.0	0.42	0.59	0.38	0.88	0.53
259	0.36	0.91	0.51	1.0	0.48	0.65	0.43	0.91	0.58
260	0.28	0.69	0.40	0.86	0.21	0.33	0.28	0.55	0.37
261	0.21	0.55	0.31	0.90	0.27	0.42	0.25	0.55	0.34
262	0.19	0.30	0.23	1.0	0.27	0.43	0.21	0.27	0.24
265	0.22	0.48	0.30	0.90	0.31	0.46	0.28	0.48	0.35
266	0.21	0.55	0.31	0.90	0.27	0.42	0.25	0.55	0.35
301	0.46	0.85	0.60	0.94	0.82	0.88	0.56	0.83	0.67
302	0.32	0.72	0.44	0.97	0.60	0.74	0.43	0.70	0.53
303	0.22	0.86	0.35	0.93	0.82	0.87	0.42	0.86	0.56
304	0.63	0.97	0.76	0.97	0.96	0.97	0.72	0.97	0.83
H-mean	0.61	0.90	0.71	0.99	0.79	0.88	0.70	0.89	0.77

References

L. Amgoud and C. Cayrol. On the acceptability of arguments in preference-based argumentation. In *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, pages 1–7, San Francisco, California, 1998. Morgan Kaufmann.

L. Amgoud and C. Cayrol. Inferring from inconsistency in preference-based argumentation frameworks. *Journal of Automated Reasoning*, 29(2):125–169, 2002a. ISSN 0168-7433.

L. Amgoud and C. Cayrol. A reasoning model based on the production of acceptable arguments. *Annals of Mathematics and Artificial Intelligence*, 34(1-3):197–215, 2002b. ISSN 1012-2443.

Y. An, A. Borgida, and J. Mylopoulos. Discovering the semantics of relational tables through mappings. *Journal on Data Semantics VII*, 4244/2006:1–32, 2006.

F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, and P. F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications*, 2003. Cambridge University Press. ISBN 0-521-78176-0.

T. Bach and R. Dieng-Kuntz. Measuring similarity of elements in owl ontologies. In *Proceedings of AAAI Workshop on Contexts and Ontologies*, pages 96–99, 2005.

T. Bach, R. Dieng-Kuntz, and F. Gandon. On ontology matching problems – for building a corporate semantic web in a multi-communities organization. In *Proceedings of the 6th International Conference on Enterprise Information Systems*, volume IV, pages 236–243, 2004.

S. C. Bailin and W. Truszkowski. Ontology negotiation between intelligent information agents. *Knowledge Engineering Review*, 17(1):7–19, 2002. ISSN 0269-8889. doi: http://dx.doi.org/10.1017/S0269888902000292.

M. Barbuceanu and W.-K. Lo. A multi-attribute utility theoretic negotiation architecture for electronic commerce. In *Proceedings of the 4th international Conference on Autonomous agents*, pages 239–246, New York, NY, USA, 2000. ACM. ISBN 1-58113-230-1. doi: http://doi.acm.org/10.1145/336595.337460.

J. Barwise and J. Seligman. *Information Flow: The Logic of Distributed Systems*. Cambridge University Press, New York, NY, USA, 1997. ISBN 0-521-58386-1.

C. Batini, M. Lenzerini, and S. Navathe. A comparative analysis of methodologies for database schema integration. *ACM Computing Surveys*, 18(4):323–364, 1986. ISSN 0360-0300. doi: http://doi.acm.org/10.1145/27633.27634.

M. Beer, M. D'inverno, M. Luck, N. Jennings, C. Preist, and M. Schroeder. Negotiation in multi-agent systems. *Knowledge Engineering Review*, 14(3):285–289, 1999. ISSN 0269-8889. doi: http://dx.doi.org/10.1017/S0269888999003021.

T. Bench-Capon. Value-based argumentation frameworks, 2002. URL `http://www.citebase.org/abstract?id=oai:arXiv.org:cs/0207059`.

T. Bench-Capon. Persuasion in practical argument using value-based argumentation frameworks. *Journal of Logic and Computation*, 13(3):429–448, 2003.

T. Bench-Capon. Ontologies and legal knowledge-based systems development. Technical report, Department of Computer Science, University of Liverpool, 2005.

T. Bench-Capon and P. Dunne. Value-based argumentation frameworks. Technical report, Department of Computer Science, University of Liverpool, 2002.

M. Benerecetti, P. Bouquet, and C. Ghidini. On the dimensions of context dependence: Partiality, approximation, and perspective. In *Proceedings of the Third International and Interdisciplinary Conference on Modeling and Using Context*, pages 59–72, London, UK, 2001. Springer-Verlag. ISBN 3-540-42379-6.

J. Berlin and A. Motro. Database schema matching using machine learning with feature selection. In *Proceedings of the 14th International Conference on Advanced Information Systems Engineering*, pages 452–466, London, UK, 2002. Springer-Verlag. ISBN 3-540-43738-X.

P. Bernstein and E. Rahm. Data warehouse scenarios for model management. In *Conceptual Modeling*, volume 1920/2000 of *Lecture Notes in Computer Science*, pages 57–109. Springer Berlin / Heidelberg, 2000. ISBN 978-3-540-41072-0. doi: 10.1007/3-540-45393-8-1.

R.-J. Beun, R. M. van Eijk, and H. Prust. Ontological feedback in multiagent systems. In *Proceedings of the 3rd International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 110–117, Washington, DC, USA, 2004. IEEE Computer Society. ISBN 1-58113-864-4. doi: http://dx.doi.org/10.1109/AAMAS.2004.204.

J. Bigham and L. Du. Cooperative negotiation in a multi-agent system for real-time load balancing of a mobile cellular network. In *Proceedings of the 2nd International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 568–575, New York, NY, USA, 2003. ACM. ISBN 1-58113-683-8. doi: http://doi.acm.org/10.1145/860575.860666.

A. Bilke and F. Naumann. Schema matching using duplicates. In *Proceedings of the 21st International Conference on Data Engineering*, pages 69–80, Washington, DC, USA, 2005. IEEE Computer Society. ISBN 0-7695-2285-8. doi: http://dx.doi.org/10.1109/ICDE.2005.126.

P. Bouquet, M. Ehrig, J. Euzenat, E. Franconi, P. Hitzler, M. Krotzsch, L. Serafini, G. Stamou, Y. Sure, and S. Tessaris. Specification of a common framework for characterizing alignment. Technical Report Knowledge Web Deliverable 2.2.1v2, University of Karlsruhe, 2004.

P. Bouquet, L. Serafini, S. Zanobini, and S. Sceffer. Bootstrapping semantics on the web: Meaning elicitation from schemas. In *Proceedings of the 15th international conference on World Wide Web*, pages 505–512. ACM Press, 2006. doi: http://doi.acm.org/10.1145/1135777.1135851.

F.-E. Calvier and C. Reynaud. Ontology matching supported by query answering in a p2p system. In *Proceedings of the 7th International Conference on Ontologies, DataBases, and Applications of Semantics*, Monterrey, Mexico, 2008.

S. Castano, V. D. Antonellis, and S. D. C. di Vimercati. Global viewing of heterogeneous data sources. *IEEE Transaction on Knowledge and Data Engineering*, 13(2):277–297, 2001. ISSN 1041-4347. doi: http://dx.doi.org/10.1109/69.917566.

S. Castano, A. Ferrara, and G. Messa. Hmatch results for oaei 2006. In P. Shvaiko, J. Euzenat, N. Noy, H. Stuckenschmidt, R. Benjamins, and M. Uschold, editors, *Proceeding of the 1st International Workshop on Ontology Matching*, volume 225 of *CEUR Workshop Proceedings*, Athens, Georgia, USA, 2006. CEUR-WS.org.

B. Chandrasekaran, J. R. Josephson, and R. Benjamins. What are ontologies, and why do we need them? *IEEE Intelligent Systems*, 14(1):20–26, 1999. ISSN 1541-1672. doi: http://dx.doi.org/10.1109/5254.747902.

N. Choi, I.-Y. Song, and H. Han. A survey on ontology mapping. *ACM SIGMOD Record*, 35(3): 34–41, 2006. ISSN 0163-5808. doi: http://doi.acm.org/10.1145/1168092.1168097.

C. Curino, G. Orsi, and L. Tanca. X-som: A flexible ontology mapper. In *Proceedings of the 18th International Conference on Database and Expert Systems Applications*, pages 424–428, Washington, DC, USA, 2007a. IEEE Computer Society. ISBN 0-7695-2932-1. doi: http://dx.doi.org/10.1109/DEXA.2007.175.

C. Curino, G. Orsi, and L. Tanca. X-som results for oaei 2007. In *Proceedings of the 2nd Ontology Matching Workshop*, pages 276–285, Busan, Korea, 2007b.

R. Dhamankar, Y. Lee, A. Doan, A. Halevy, and P. Domingos. imap: discovering complex semantic matches between database schemas. In *Proceedings of the 2004 ACM SIGMOD international conference on Management of data*, pages 383–394, New York, NY, USA, 2004. ACM. ISBN 1-58113-859-8. doi: http://doi.acm.org/10.1145/1007568.1007612.

H.-H. Do. *Schema Matching and Mapping-based Data Integration: Architecture, Approaches and Evaluation*. VDM Verlag Dr. Müller, Saarbrucken, 2006.

H.-H. Do and E. Rahm. Coma: a system for flexible combination of schema matching approaches. In *Proceedings of the 28th international conference on Very Large Data Bases*, pages 610–621. VLDB Endowment, 2002.

A. Doan, P. Domingos, and A. Y. Halevy. Reconciling schemas of disparate data sources: A machine-learning approach. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 509–520, New York, NY, USA, 2001. ACM. ISBN 1-58113-332-4. doi: http://doi.acm.org/10.1145/375663.375731.

A. Doan, J. Madhavan, P. Domingos, and A. Halevy. Ontology matching: A machine learning approach. In S. Staab and R. Studer, editors, *Handbook on Ontologies in Information Systems*, chapter 18, pages 385–404. Springer-Verlag, 2004.

S. Dumais, M. Banko, E. Brill, J. Lin, and A. Ng. Web question answering: is more always better? In *Proceedings of the 25th Annual International ACM Conference on Research and Development in Information Retrieval*, pages 291–298, New York, NY, USA, 2002. ACM. ISBN 1-58113-561-0. doi: http://doi.acm.org/10.1145/564376.564428.

P. Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n–person games. *Artificial Intelligence*, 77(2):321–357, 1995.

M. Ehrig. *Ontology Alignment: Bridging the Semantic Gap*, volume 4 of *Semantic Web And Beyond Computing for Human Experience*. Springer, 2007. ISBN 978-0-387-36501-5.

M. Ehrig and S. Staab. Qom - quick ontology mapping. In *The Semantic Web - ISWC 2004*, volume 3298/2004 of *Lecture Notes in Computer Science*, pages 683–697. Springer Berlin / Heidelberg, 2004.

M. Ehrig and Y. Sure. Ontology mapping - an integrated approach. In *European Semantic Web Symposium*, volume 3053/2004 of *Lecture Notes in Computer Science*, pages 76–91. Springer Berlin / Heidelberg, 2004. ISBN 978-3-540-21999-6. doi: 10.1007/b97867.

M. Elvang-Gand A. Hunter. Argumentative logics: reasoning with classically inconsistent information. *Data Knowledge Engineering*, 16(2):125–145, 1995. ISSN 0169-023X. doi: http://dx.doi.org/10.1016/0169-023X(95)00013-I.

D. W. Embley, L. Xu, and Y. Ding. Automatic direct and indirect schema mapping: Experiences and lessons learned. *SIGMOD Record*, 33(4):14–19, 2004. ISSN 0163-5808. doi: http://doi.acm.org/10.1145/1041410.1041413.

J. Euzenat. Towards a principled approach to semantic interoperability. In A. Pérez, M. Gruninger, H. Stuckenschmidt, and M. Uschold, editors, *IJCAI Workshop on Ontologies and Information Sharing*, pages 19–25, Seattle, WA US, 2001.

J. Euzenat and P. Shvaiko. *Ontology Matching*. Springer-Verlag, Heidelberg (DE), 2007. ISBN 3-540-49611-4.

J. Euzenat and P. Valtchev. Similarity-based ontology alignment in OWL-Lite. In *Proceedings of the European Conference on Artificial Intelligence*, volume 16, pages 333–337, 2004.

J. Euzenat, H. Stuckenschmidt, and M. Yatskevich. Introduction to the ontology alignment evaluation 2005. In B. Ashpole, M. Ehrig, J. Euzenat, and H. Stuckenschmidt, editors, *Workshop on Integrating Ontologies*, volume 156 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2005.

J. Euzenat, M. Mochol, P. Shvaiko, H. Stuckenschmidt, O. Sváb, V. Svátek, W. R. van Hage, and M. Yatskevich. Results of the ontology alignment evaluation initiative 2006. In P. Shvaiko, J. Euzenat, N. F. Noy, H. Stuckenschmidt, V. R. Benjamins, and M. Uschold, editors, *Proceedings of the 1st International Workshop on Ontology Matching*, volume 225 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2006.

J. Euzenat, A. Isaac, C. Meilicke, P. Shvaiko, H. Stuckenschmidt, O. Sváb, V. Svátek, W. R. van Hage, and M. Yatskevich. Results of the ontology alignment evaluation initiative 2007. In P. Shvaiko, J. Euzenat, F. Giunchiglia, and B. He, editors, *Proceedings of the 2nd International Workshop on Ontology Matching*, volume 304 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2007.

P. Faratin, C. Sierra, and N. Jennings. Negotiation decision functions for autonomous agents. *Robotics and Autonomous Systems*, 24(3):159–182, 1998. doi: 10.1016/S0921-8890(98)00029-3.

S. S. Fatima, M. Wooldridge, and N. R. Jennings. An agenda-based framework for multi-issue negotiation. *Artificial Intelligence*, 152(1):1–45, 2004. ISSN 0004-3702. doi: http://dx.doi.org/10.1016/S0004-3702(03)00115-2.

D. Fensel. *Ontologies: A Silver Bullet for Knowledge Management and Electronic Commerce.* Springer-Verlag New York, Inc., 2nd edition, 2003.

D. Fensel, I. Horrocks, F. van Harmelen, S. Decker, M. Erdmann, and M. Klein. OIL in a nutshell. In R. Dieng, editor, *Knowledge Acquisition, Modeling and Management*, number 1937 in Lecture Notes in Artificial Intelligence, pages 1–16. Springer-Verlag, 2000.

D. Fensel, H. Lausen, A. Polleres, J. de Bruijn, M. Stollberg, D. Roman, and J. Domingue. *Enabling Semantic Web Services: The Web Service Modeling Ontology.* Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2007. ISBN 978-3-540-34519-0.

A. Gal, G. Modica, and H. Jamil. Ontobuilder: Fully automatic extraction and consolidation of ontologies from web sources. In *Proceedings of the 20th International Conference on Data Engineering*, pages 853–862, Washington, DC, USA, 2004. IEEE Computer Society. ISBN 0-7695-2065-0.

N. Gatti and F. Amigoni. A cooperative negotiation protocol for physiological model combination. In *Proceedings of the 3th International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 655–662, Washington, DC, USA, 2004. IEEE Computer Society. ISBN 1-58113-864-4. doi: http://dx.doi.org/10.1109/AAMAS.2004.10.

F. Giunchiglia and P. Shvaiko. Semantic matching. *Knowledge Engineering Review*, 18(3):265–280, 2003. ISSN 0269-8889. doi: http://dx.doi.org/DOI:10.1017/S0269888904000074.

F. Giunchiglia, P. Shvaiko, and M. Yatskevich. S-match: An algorithm and an implementation of semantic matching. In *The Semantic Web: Research and Applications*, volume 3053/2004 of *Lecture Notes in Computer Science*, pages 61–75. Springer Berlin / Heidelberg, 2004a. ISBN 978-3-540-21999-6.

F. Giunchiglia, P. Shvaiko, and M. Yatskevich. S-match: An algorithm and an implementation of semantic matching. In *European Semantic Web Symposium*, volume 3053/2004 of *Lecture Notes in Computer Science*, pages 61–75. Springer Verlag / Heidelberg, 2004b. ISBN 978-3-540-21999-6. doi: 10.1007/b97867.

F. Giunchiglia, P. Shvaiko, and M. Yatskevich. Semantic schema matching. In *OTM Conferences*, volume 3761 of *Lecture Notes in Computer Science*, pages 347–365. Springer Berlin, 2005. ISBN 3-540-29736-7.

A. Gomez-Perez and O. Corcho. Ontology languages for the semantic web. *Intelligent Systems*, 17 (1):54–60, 2002. doi: http://dx.doi.org/10.1017/S0269888998001040.

S. Green, L. Hurst, B. Nangle, P. Cunningham, F. Somers, and R. Evans. Software agents: A review. Technical Report TCD-CS-1997-06, Trinity College, Dublin, Ireland, 1997.

T. Gruber. A translation approach to portable ontology specifications. *Knowledge Acquisition*, 5(2): 199–220, 1993. ISSN 1042-8143. doi: http://dx.doi.org/10.1006/knac.1993.1008.

T. Gruber. Towards principles for the design of ontologies used for knowledge sharing. *International Journal Human Computer Studies*, 43(5-6):907–928, 1995.

N. Guarino. Formal ontology and information systems. In N. Guarino, editor, *Proceedings of the First International Conference on Formal Ontologies in Information Systems*, pages 3–15, Amsterdam, The Netherlands, 1998. IOS Press.

N. Guarino and P. Giaretta. Ontologies and knowledge bases: Towards a terminological clarification. In N. Mars, editor, *Towards Very Large Knowledge Bases: Knowledge Building and Knowledge Sharing*, pages 25–32. IOS Press, 1995.

R. H. Guttman and P. Maes. Cooperative vs. competitive multi-agent negotiations in retail electronic commerce. In *Proceedings of the Second International Workshop on Cooperative Information Agents II, Learning, Mobility and Electronic Commerce for Information Discovery on the Internet*, pages 135–147, London, UK, 1998. Springer-Verlag. ISBN 3-540-64676-0.

L. M. Haas, M. A. Hernández, H. Ho, L. Popa, and M. Roth. Clio grows up: From research prototype to industrial tool. In *Proceedings of the 2005 ACM SIGMOD international conference on Management of data*, pages 805–810, New York, NY, USA, 2005. ACM. ISBN 1-59593-060-4. doi: http://doi.acm.org/10.1145/1066157.1066252.

F. Hakimpour and A. Geppert. Resolving semantic heterogeneity in schema integration. In *Proceedings of the international conference on Formal Ontology in Information Systems*, pages 297–308, New York, NY, USA, 2001. ACM. ISBN 1-58113-377-4. doi: http://doi.acm.org/10.1145/505168.505196.

A. Halevy, N. Ashish, D. Bitton, M. Carey, D. Draper, J. Pollock, A. Rosenthal, and V. Sikka. Enterprise information integration: successes, challenges and controversies. In *Proceedings of the 2005 ACM SIGMOD international conference on Management of data*, pages 778–787, New York, NY, USA, 2005. ACM. ISBN 1-59593-060-4. doi: http://doi.acm.org/10.1145/1066157.1066246.

M. A. Hernández, R. J. Miller, and L. M. Haas. Clio: A semi-automatic tool for schema mapping. *SIGMOD Rec.*, 30(2):607, 2001. ISSN 0163-5808. doi: http://doi.acm.org/10.1145/376284.375767.

M. Horridge, H. Knublauch, A. Rector, R. Stevens, and C. Wroe. A practical guide to building owl ontologies using the protege-owl plugin and co-ode tools edition 1.0. Technical report, The University Of Manchester, 2004.

I. Horrocks and P. Patel-Schneider. Reducing owl entailment to description logic satisfiability. In *Proceedings of the 2nd International Semantic Web Conference*, volume 2870/2003 of *Lecture Notes in Computer Science*, pages 17–29. Springer Berlin / Heidelberg, 2003. ISBN 978-3-540-20362-9. doi: 10.1007/b14287.

E. Hovy. Combining and standardizing large-scale, practical ontologies for machine translation and other uses. In *First International Conference on Language Resources and Evaluation*, pages 535–542, 1998.

W. Hu, N. Jian, Y. Qu, and Y. Wang. Gmo: A graph matching for ontologies. In B. Ashpole, M. Ehrig, J. Euzenat, and H. Stuckenschmidt, editors, *K-CAP Integrating Ontologies*, volume 156 of *CEUR Workshop Proceedings*, pages 43–50. CEUR-WS.org, 2005.

W. Hu, G. Cheng, D. Zheng, X. Zhong, and Y. Qu. The results of falcon-ao in the oaei 2006 campaign. In P. Shvaiko, J. Euzenat, N. F. Noy, H. Stuckenschmidt, V. R. Benjamins, and M. Uschold, editors, *Ontology Matching*, volume 225 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2006.

J. Hubner. *Um Modelo de Reorganização de Sistemas Multiagentes*. PhD thesis, Escola Politécnica da Universidades de São Paulo, Departamento de Engenharia da Computação e Sistemas Digitais, 2003.

J. F. Hübner, a. S. Jaime Sim and O. Boissier. A model for the structural, functional, and deontic specification of organizations in multiagent systems. In *Proceedings of the 16th Brazilian Symposium on Artificial Intelligence*, pages 118–128, London, UK, 2002. Springer-Verlag. ISBN 3-540-00124-7.

R. Ichise, M. Hamasaki, and ideaki Takeda. Discovering relationships among catalogs. In *Discovery Science*, Lecture Notes in Computer Science, pages 371–379. Springer Berlin / Heidelberg, 2004. ISBN 978-3-540-23357-2.

A. Isaac, H. Matthezing, L. van der Meij, S. Schlobach, S. Wang, and C. Zinn. Putting ontology alignment in context: usage scenarios, deployment and evaluation in a library case. In *ESWC 2008*, Tenerife, Spain, 2008a.

A. Isaac, L. van der Meij, S. Schlobach, and S. Wang. An empirical study of instance-based ontology matching. In *The Semantic Web Conference 2007*, volume 4825/2008 of *Lecture Notes in Computer Science*, Busan, Korea, 2008b. Springer Berlin / Heidelberg.

Y. Jean-Mary and M. Kabuka. Asmov: Ontology alignment with semantic validation. In *Joint SWDB-ODBIS Workshop on Semantics, Ontologies, Databases*, pages 15–20, Vienna, Austria, 2007a.

Y. Jean-Mary and M. Kabuka. Asmov results for oaei 2007. In *Proceedings of the 2nd Ontology Matching Workshop*, pages 141–150, Busan, Korea, 2007b.

N. Jennings, P. Faratin, A. Lomuscio, S. Parsons, M. Wooldridge, and C. Sierra. Automated negotiation: Prospects methods and challenges. *Group Decision and Negotiation*, 10(2):199–215, 2001.

N. R. Jennings, S. Parsons, P. Noriega, and C. Sierra. On argumentation-based negotiation. In *International Workshop on Multi-Agent Systems*. MIT, MIT, 1998.

N. Jian, W. Hu, G. Cheng, and Y. Qu. Falconao: Aligning ontologies with falcon. In B. Ashpole, M. Ehrig, J. Euzenat, and H. Stuckenschmidt, editors, *K-CAP Integrating Ontologies*, volume 156 of *CEUR Workshop Proceedings*, pages 85–91. CEUR-WS.org, 2005.

T. Joachims. *Learning to Classify Text Using Support Vector Machines – Methods, Theory and Algorithms*, volume 668 of *The Springer International Series in Engineering and Computer Science*. Kluwer Academic Publishers/Springer, 2002. ISBN 978-0-7923-7679-8.

S. Jun-feng, Z. Wei-ming, X. Wei-dong, L. Guo-hui, and X. Zhen-ning. Ontology-based information retrieval model for the semantic web. In *Proceedings of the 2005 IEEE International Conference on e-Technology, e-Commerce and e-Service*, pages 152–155, Washington, DC, USA, 2005. IEEE Computer Society. ISBN 0-7695-2274-2. doi: http://dx.doi.org/10.1109/EEE.2005.98.

Y. Kalfoglou. *Deploying Ontologies in Software Design*. PhD thesis, Department of Artificial Intelligence, University of Edinburgh, June 2000.

Y. Kalfoglou and M. Schorlemmer. If-map: an ontology mapping method based on information flow theory. *Journal on Data Semantics*, 2800/2003:98–127, 2003a.

Y. Kalfoglou and M. Schorlemmer. Ontology mapping: The state of the art. *Knowledge Engineering Review*, 18(1):1–31, 2003b. ISSN 0269-8889. doi: http://dx.doi.org/10.1017/S0269888903000651.

J. Kang and J. F. Naughton. On schema matching with opaque column names and data values. In *Proceedings of the 2003 ACM SIGMOD international conference on Management of data*, pages 205–216, New York, NY, USA, 2003. ACM. ISBN 1-58113-634-X.

M. Kifer, G. Lausen, and J. Wu. Logical foundations of object-oriented and frame-based languages. *Journal of the ACM*, 42(4):741–843, 1995.

J. Kim, M. Jang, Y.-G. Ha, J.-C. Sohn, and S. J. Lee. Moa: Owl ontology merging and alignment tool for the semantic web. In *Proceedings of the 18th international conference on Innovations in Applied Artificial Intelligence*, pages 722–731, London, UK, 2005. Springer-Verlag. ISBN 3-540-26551-1.

C.-C. Kiu and C.-S. Lee. Ontology mapping and merging through ontodna for learning object reusability. *Journal of Educational Technology and Society*, 9(3):27–42, 2006.

C.-C. Kiu and C.-S. Lee. Ontodna: Ontology alignment results for oaei 2007. In *Proceedings of the 2nd Ontology Matching Workshop*, pages 196–204, Busan, Korea, 2007.

M. Klein. Combining and relating ontologies: An analysis of problems and solutions. In A. Pérez, M. Gruninger, H. Stuckenschmidt, and M. Uschold, editors, *IJCAI Workshop on Ontologies and Information Sharing*, Seattle, WA, USA, 2001.

K. Kotis, G. Vouros, and K. Stergiou. Towards automatic merging of domain ontologies: The hcone-merge approach. *Journal of Web Semantic*, 4(1):60–79, 2005.

S. Kraus. Negotiation and cooperation in multi-agent environments. *Artificial Intelligence*, 94(1-2): 79–97, 1997. ISSN 0004-3702. doi: http://dx.doi.org/10.1016/S0004-3702(97)00025-8.

S. Kraus, O. Shehory, and G. Taase. Coalition formation with uncertain heterogeneous information. In *Proceedings of the 2nd International joint Conference on Autonomous Agents and Multiagent Systems*, pages 1–8, New York, NY, USA, 2003. ACM. ISBN 1-58113-683-8. doi: http://doi.acm.org/10.1145/860575.860577.

M. Lacher and G. Groh. Facilitating the exchange of explicit knowledge through ontology mappings. In *Proceedings of the Fourteenth International Florida Artificial Intelligence Research Society Conference*, pages 305–309. AAAI Press, 2001. ISBN 1-57735-133-9.

L. Laera, V. Tamma, J. Euzenat, T. Bench-Capon, and T. Payne. Reaching agreement over ontology alignments. In *The Semantic Web - ISWC 2006*, volume 4273/2006 of *Lecture Notes in Computer Science*, pages 371–384. Springer Berlin / Heidelberg, 2006. ISBN 978-3-540-49029-6. doi: 10.1007/11926078.

L. Laera, I. Blacoe, V. Tamma, T. Payne, J. Euzenat, and T. Bench-Capon. Argumentation over ontology correspondences in mas. In *Proceedings of the 6th International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 1–8, New York, NY, USA, 2007. ACM. ISBN 978-81-904262-7-5. doi: http://doi.acm.org/10.1145/1329125.1329400.

P. Lambrix and H. Tan. Sambo: A system for aligning and merging biomedical ontologies. *Journal of Web Semantic*, 4(3):196–206, 2006. ISSN 1570-8268. doi: http://dx.doi.org/10.1016/j.websem.2006.05.003.

M. L. Lee, L. H. Yang, W. Hsu, and X. Yang. Xclust: clustering xml schemas for effective integration. In *Proceedings of the 11th International Conference on Information and Knowledge Management*, pages 292–299, New York, NY, USA, 2002. ACM. ISBN 1-58113-492-4.

V. Levenshtein. Binary Codes Capable of Correcting Deletions and Insertions and Reversals. *Soviet Physics Doklady*, 10(8):707–710, 1966.

W.-S. Li and C. Clifton. Semint: A tool for identifying attribute correspondences in heterogeneous databases using neural networks. *Data Knowl. Eng.*, 33(1):49–84, 2000. ISSN 0169-023X. doi: http://dx.doi.org/10.1016/S0169-023X(99)00044-0.

A. Lomuscio, M. Wooldridge, and N. Jennings. A classification scheme for negotiation in electronic commerce. In *Agent Mediated Electronic Commerce: The European AgentLink Perspective*, pages 19–33, London, UK, 2001. Springer-Verlag. ISBN 3-540-41671-4.

R. MacGregor. Inside the loom description classifier. *ACM SIGART Bulletin*, 2(3):88–92, 1991. ISSN 0163-5719. doi: http://doi.acm.org/10.1145/122296.122309.

J. Madhavan, P. Bernstein, and E. Rahm. Generic schema matching with cupid. In *Proceedings of the 27th International Conference on Very Large Data Bases*, pages 49–58, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc. ISBN 1-55860-804-4.

J. Madhavan, P. A. Bernstein, A. Doan, and A. Halevy. Corpus-based schema matching. In *Proceedings of the 21st International Conference on Data Engineering*, pages 57–68, Washington, DC, USA, 2005. IEEE Computer Society. ISBN 0-7695-2285-8. doi: http://dx.doi.org/10.1109/ICDE.2005.39.

A. Maedche and S. Staab. Measuring similarity between ontologies. In *Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management. Ontologies and the Semantic Web*, pages 251–263, London, UK, 2002. Springer-Verlag. ISBN 3-540-44268-5.

A. Maedche, B. Motik, N. Silva, and R. Volz. Mafra - a mapping framework for distributed ontologies. In *Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management. Ontologies and the Semantic Web*, pages 235–250, London, UK, 2002. Springer-Verlag. ISBN 3-540-44268-5.

R. Mailler, V. Lesser, and B. Horling. Cooperative negotiation for soft real-time distributed resource allocation. In *Proceedings of the 2nd International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 576–583, New York, NY, USA, 2003. ACM. ISBN 1-58113-683-8. doi: http://doi.acm.org/10.1145/860575.860667.

M. Mao and Y. Peng. Prior system: Results for oaei 2006. In *Proceedings of the 1st Ontology Matching Workshop*, Georgia, USA, 2006.

M. Mao and Y. Peng. The prior+: Results for oaei campaign 2007. In *Proceedings of the 2nd Ontology Matching Workshop*, pages 219–226, Busan, Korea, 2007.

M. Mao, Y. Peng, and M. Spring. A profile propagation and information retrieval based ontology mapping approach. In *Proceedings of the 3th International Conference on Semantics, Knowledge and Grid*, pages 164–169, Washington, DC, USA, 2007. IEEE Computer Society. ISBN 0-7695-3007-9. doi: http://dx.doi.org/10.1109/SKG.2007.172.

C. Masolo, S. Borgo, A. Gangemi, N. Guarino, A. Oltramari, and L. Schneider. The wonderweb library of foundational ontologies. Technical Report D17, ISTC-CNR, 2003.

S. Melnik, H. Garcia-Molina, and E. Rahm. Similarity flooding: a versatile graph matching algorithm and its application to schema matching. In *Proceedings of the 18th International Conference on Data Engineering*, pages 117–128, Washington, DC, USA, 2002. IEEE Computer Society.

A. Miles and S. Bechhofer. Skos reference. Technical report, W3C, January 25 2008. URL `http://www.w3.org/TR/skos-primer/`.

R. J. Miller, L. M. Haas, and M. A. Hernández. Schema mapping as query discovery. In *Proceedings of the 26th International Conference on Very Large Data Bases*, pages 77–88, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc. ISBN 1-55860-715-3.

T. Milo and S. Zohar. Using schema matching to simplify heterogeneous data translation. In *Proceedings of the 24rd International Conference on Very Large Data Bases*, pages 122–133, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc. ISBN 1-55860-566-5.

P. Mitra, G. Wiederhold, and J. Jannink. Semi-automatic integration of knowledge sources. In *Proceedings of the Second International Conference on Information Fusion*, pages 572–581, 1999.

P. Mitra, G. Wiederhold, and M. Kersten. A graph-oriented model for articulation of ontology interdependencies. In *Advances in Database Technology*, volume 1777/2000 of *Lecture Notes in Computer Science*, pages 86–100. Springer Berlin / Heidelberg, 2000. ISBN 978-3-540-67227-2.

P. Mitra, N. Noy, and A. Jaiswal. Ontology mapping discovery with uncertainty. In *Proceedings of 4th International Semantic Web Conference*, pages 537–545. Springer-Verlag, 2005.

G. Modica, A. Gal, and H. M. Jamil. The use of machine-generated ontologies in dynamic information seeking. In *Proceedings of the 9th International Conference on Cooperative Information Systems*, volume 2172/2001 of *Lecture Notes in Computer Science*, pages 433–447. Springer Berlin / Heidelberg, 2001. ISBN 978-3-540-42524-3.

M. Nagy, M. Vargas-Vera, and E. Motta. DSSim – managing uncertainty on the semantic web. In *2nd International Workshop on Ontology Matching*, Busan, Korea, 2007.

I. Niles and A. Pease. Towards a standard upper ontology. In *Proceedings of the international conference on Formal Ontology in Information Systems*, pages 2–9, New York, NY, USA, 2001. ACM. ISBN 1-58113-377-4. doi: http://doi.acm.org/10.1145/505168.505170.

H. Nottelmann and U. Straccia. splmap: A probabilistic approach to schema matching. In D. E. Losada and J. M. Fernández-Luna, editors, *Proceedings of the 27th European Conference on Information Retrieval Research*, volume 3408 of *Lecture Notes in Computer Science*, pages 81–95. Springer-Verlag, 2005.

N. Noy and M. Klein. Ontology evolution: Not the same as schema evolution. *Knowledge and Information Systems*, 6(4):428–440, July 2004.

N. Noy and D. L. McGuiness. Ontology development 101: A guide to creating your first ontology. Technical report, Stanford University, 2001.

N. Noy and M. Musen. Anchor-prompt: Using non-local context for semantic matching. In A. Pérez, M. Gruninger, H. Stuckenschmidt, and M. Uschold, editors, *IJCAI Workshop on Ontologies and Information Sharing*, pages 63–70, Seattle, WA US, 2001.

Óscar Corcho and A. Gómez-Pérez. A roadmap to ontology specification languages. In *Proceedings of the 12th European Workshop on Knowledge Acquisition, Modeling and Management*, pages 80–96, London, UK, 2000. Springer-Verlag. ISBN 3-540-41119-4.

R. Ossewaarde. Simple library thesaurus alignment with SILAS. In *2nd International Workshop on Ontology Matching*, Busan, Korea, 2007.

L. Palopoli, L. Pontieri, G. Terracina, and D. Ursino. Intensional and extensional integration and abstraction of heterogeneous databases. *Data Knowledge Engineering*, 35(3):201–237, 2000. ISSN 0169-023X. doi: http://dx.doi.org/10.1016/S0169-023X(00)00028-8.

L. Palopoli, D. Saccà, G. Terracina, and D. Ursino. Uniform techniques for deriving similarities of objects and subschemes in heterogeneous databases. *IEEE Transaction on Knowledge and Data Engineering*, 15(2):271–294, 2003. ISSN 1041-4347. doi: http://dx.doi.org/10.1109/TKDE.2003.1185834.

S. Parsons and N. Jennings. Negotiation through argumentation-A preliminary report. In *Proceedings of the Second International Conference Multi-Agent Systems*, pages 267–274, Kyoto, Japan, 1996a.

S. Parsons, C. Sierra, and N. Jennings. Agents that reason and negotiate by arguing. *Journal of Logic and Computation*, 8(3):261–292, 1998.

S. D. Parsons and N. Jennings. Negotiation through argumentation-A preliminary report. In *Proceedings of the Second International Conference Multi-Agent Systems*, pages 267–274, 1996b.

H. S. Pinto, A. Gomez-Perez, and J. Martins. Some issues on ontology integration. In *Proceedings of the IJCAI Workshop on Ontologies and Problem Solving Methods: Lessons Learned and Future Trends*, pages 7–12, Stockholm, Sweden, 1999.

H. Prakken and G. Sartor. Argument-based extended logic programming with defeasible priorities. *Journal Applied Non-Classical Logics*, 7(1):25–75, 1997.

H. Prakken and G. Vreeswijk. Logical systems for defeasible argumentation. *Handbook of Philosophical Logic*, 4:219–318, 2000.

Y. Qu, W. Hu, and G. Cheng. Constructing virtual documents for ontology matching. In *Proceedings of the 15th International Conference on World Wide Web*, pages 23–31, New York, NY, USA, 2006. ACM. ISBN 1-59593-323-9. doi: http://doi.acm.org/10.1145/1135777.1135786.

E. Rahm and P. Bernstein. A survey of approaches to automatic schema matching. *International Journal on Very Large Data Bases*, 10(4):334–350, 2001. ISSN 1066-8888. doi: http://dx.doi.org/10.1007/s007780100057.

J. S. Rosenschein and G. Zlotkin. *Rules of Encounter: Designing Conventions for Automated Negotiation among Computers*. MIT Press, Cambridge, MA, USA, 1994. ISBN 0-262-18159-2.

J. Sabater, C. Sierra, S. Parsons, and N. R. Jennings. Using multi-context systems to engineer executable agents. In *Proceedings of the 6th International Workshop on Intelligent Agents VI, Agent Theories, Architectures, and Languages*, pages 260–276, London, UK, 2000. Springer-Verlag. ISBN 3-540-67200-1.

M. Sabou, V. Lopez, and E. Motta. Ontology selection for the real semantic web: How to cover the queen's birthday dinner? In *Proceedings of the 15th International Conference on Knowledge Engineering and Knowledge Management*, volume 4248 of *Lecture Notes in Computer Science*, pages 96–111. Springer Berlin, 2006.

T. Sandholm. Agents in electronic commerce: Component technologies for automated negotiation and coalition formation. *Autonomous Agents and Multi-Agent Systems*, 3(1):73–96, 2000. ISSN 1387-2532. doi: http://dx.doi.org/10.1023/A:1010038012192.

T. W. Sandholm. Distributed rational decision making. *Multiagent Systems: a Modern Approach to Distributed Artificial Intelligence*, pages 201–258, 1999.

M. Schorlemmer. Duality in knowledge sharing. In *Proceedings Seventh International Symposium on Artificial Intelligence and Mathematics*, 2002.

P. Shvaiko. A classification of schema-based matching approaches. Technical Report DIT-04-093, Informatica e Telecomunicazioni, University of Trento, 2004.

P. Shvaiko and J. Euzenat. A survey of schema-based matching approaches. In *Journal on Data Semantics IV*, volume 3730/2005 of *Lecture Notes in Computer Science*, pages 146–171. Springer Berlin, 2005. doi: 10.1007/11603412-5.

C. Sierra, P. Faratin, and N. R. Jennings. A service-oriented negotiation model between autonomous agents. In *Collaboration between Human and Artificial Societies, Coordination and Agent-Based Distributed Computing*, pages 201–219, London, UK, 1999. Springer-Verlag. ISBN 3-540-66930-2.

N. Silva, P. Maio, and J. Rocha. An approach to ontology mapping negotiation. In *Proceedings of the Third International Conference on Knowledge Capture Workshop on Integrating Ontologies*, Banff, Canada, 2005.

R. Smith. The contract net protocol: High-level communication and control in a distributed problem solver. *Distributed Artificial Intelligence*, pages 357–366, 1988.

V. Spiliopoulos, A. Valarakos, G. Vouros, and V. Karkaletsis. Sema: Results for the ontology alignment contest oaei 2007. In *Proceedings of the 2nd Ontology Matching Workshop*, pages 244–254, Busan, Korea, 2007.

G. Stoilos, G. Stamou, and S. Kollias. A string metric for ontology alignment. In *Proceedings of the 4th International Semantic Web Conference*, volume 3729 of *Lecture Notes in Computer Science*, pages 624–637. Springer, 2005. ISBN 3-540-29754-5.

U. Straccia and R. Troncy. omap: Combining classifiers for aligning automatically owl ontologies. In A. H. H. Ngu, M. Kitsuregawa, E. J. Neuhold, J.-Y. Chung, and Q. Z. Sheng, editors, *Web Information Systems Engineering*, volume 3806/2005 of *Lecture Notes in Computer Science*, pages 133–147. Springer Berlin / Heidelberg, 2005.

G. Stumme and A. Maedche. FCA-MERGE: Bottom-up merging of ontologies. In B. Nebel, editor, *Proceedings of the 17th International Joint Conference on Artificial Intelligence*, pages 225–234. Morgan Kaufmann, 2001. ISBN 1-55860-777-3.

K. Sycara. Multiagent systems. *AI Magazine*, 10(2):79–93, 1998.

V. Tamma, M. Wooldridge, I. Blacoe, and I. Dickinson. An ontology based approach to automated negotiation. In *Revised Papers from the Workshop on Agent Mediated Electronic Commerce on Agent-Mediated Electronic Commerce IV, Designing Mechanisms and Systems*, pages 219–237, London, UK, 2002. Springer-Verlag. ISBN 3-540-00327-4.

H. Tan and P. Lambrix. Sambo results for the ontology alignment evaluation initiative 2007. In *Proceedings of the Second Ontology Matching Workshop*, pages 236–243, Busan, Korea, 2007.

J. Tang, B. Liang, J.-Z. Li, and K. Wang. Risk minimization based ontology mapping. In *Advanced Workshop on Content Computing*, volume 3309 of *Lecture Notes in Computer Science*, pages 469–480. Springer Verlag, 2004.

F. Tohme and T. Sandholm. Coalition formation processes with belief revision among bounded-rational self-interested agents. *Journal of Logic and Computation*, 9(6):793–815, 1999.

C. Trojahn, M. Moraes, P. Quaresma, and R. Vieira. Using cooperative agent negotiation for ontology mapping. In *Proceedings of the 4th European Workshop on Multi-Agent Systems (EUMAS)*, CEUR Workshop Proceedings. CEUR-WS.org.

C. Trojahn, M. Moraes, P. Quaresma, and R. Vieira. A negotiation model for composing ontology matching approaches. *Scientia*, 16:12–27, 2005.

C. Trojahn, M. Moraes, P. Quaresma, and R. Vieira. A negotiation model for ontology mapping. In *Proceedings of the IEEE/WIC/ACM international conference on Intelligent Agent Technology*, pages 762–768, Washington, DC, USA, 2006. IEEE Computer Society. ISBN 0-7695-2748-5. doi: http://dx.doi.org/10.1109/IAT.2006.20.

C. Trojahn, P. Quaresma, and R. Vieira. An extended value-based argumentation framework for ontology mapping with confidence degrees. In *Argumentation in Multi-Agent Systems: 4th International Workshop on Argumentation in Multi-Agent Systems (ArgMAs2007)*, volume 4946, pages 132–144. Springer-Verlag, Berlin, Germany, 2008a. ISBN 978-3-540-78914-7.

C. Trojahn, P. Quaresma, and R. Vieira. Conjunctive queries for ontology based agent communication in mas. In *Proceedings of the 7th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2008)*, pages 829–836, 2008b.

C. Trojahn, P. Quaresma, and R. Vieira. An argumentation framework based on confidence degrees to combine ontology mapping approaches. *International Journal of Metadata, Semantics and Ontologies (to appear)*, 2(3), 2008c.

C. Trojahn, P. Quaresma, R. Vieira, and M. Moraes. A cooperative approach for composite ontology mapping. *LNCS Journal on Data Semantic X (JoDS)*, 4900(0):237–263, 2008d. ISSN 0302-9743 (Print) 1611-3349 (Online). doi: 10.1007/978-3-540-77688-8.

M. Uschold. Knowledge level modelling: Concepts and terminology. *Knowledge Engineering Review*, 13(1):5–29, 1998. ISSN 0269-8889. doi: http://dx.doi.org/10.1017/S0269888998001040.

J. van Diggelen, R.-J. Beun, F. Dignum, R. M. van Eijk, and J.-J. Meyer. Anemone: An effective minimal ontology negotiation environment. In *Proceedings of the 5th International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 899–906, New York, NY, USA, 2006. ACM. ISBN 1-59593-303-4. doi: http://doi.acm.org/10.1145/1160633.1160794.

H. Wache, T. Vögele, U. Visser, H. Stuckenschmidt, G. Schuster, H. Neumann, and S. Hübner. Ontology-based integration of information: A survey of existing approaches. In A. Pérez, M. Gruninger, H. Stuckenschmidt, and M. Uschold, editors, *IJCAI Workshop on Ontologies and Information Sharing*, pages 108–117, Seattle, WA US, 2001.

P. Wang and B. Xu. Lily: The results for the ontology alignment contest oaei 2007. In *Proceedings of the 2nd Ontology Matching Workshop*, pages 179–185, Busan, Korea, 2007.

P. Winoto, G. McCalla, and J. Vassileva. An extended alternating-offers bargaining protocol for automated negotiation in multi-agent systems. In *8th National Conference on Artificial Intelligence*, pages 969–970, Menlo Park, CA, USA, 2002. American Association for Artificial Intelligence. ISBN 0-262-51129-0.

M. Wooldridge. *An Introduction to Multiagent Systems*. John Wiley and Sons, 2002. ISBN 047149691X.

S. Zanobini. *Semantic Coordination: the model and an application to schema matching*. PhD thesis, International Doctorate School in Information and Communication Technology, University of Trento, March 2006.

H. Zargayouna, B. Safar, and C. Reynaud. Taxomap in the oaei 2007 alignment contest. In *Proceedings of the 2nd Ontology Matching Workshop*, pages 268–275, Busan, Korea, 2007.

X. Zhang, V. Lesser, and R. Podorozhny. Multi-dimensional, multi step negotiation for task allocation in a cooperative system. *Autonomous Agents and Multi-Agent Systems*, 10(1):5–40, 2005.

G. Zlotkin and J. Rosenchein. Mechanism for automated negotiation in state oriented domains. *Journal of Artificial Intelligence Research*, 5:163–238, 1996.