



Descoberta e composição de Serviços Web Semânticos: de condições a alternativas

Cássio Prazeres (prazer@unifacs.br) - UNIFACS

Cesar Augusto Camillo Teixeira (cesar@dc.ufscar.br) – UFSCar

Maria da Graça Pimentel (mgp@icmc.usp.br) – ICMC-USP

Conferência Web W3C Brasil'09: 24/Novembro/2009



Agradecimentos

- **FINEP**
- **FAPESP**
- **CAPES**
- **CNPq**
- **RNP/MCT**

Agenda

- Introdução
- Motivação
- Modelo em grafo para composição de serviços
- Política de custos e descontos funcionais
- Algoritmo para cálculo de caminhos de custo mínimo em grafo
- Infra-estrutura e arquitetura da implementação
- Avaliação
- Considerações finais e trabalhos futuros

Introdução

- Tendências de evolução da Web
(Martin et al., 2007; McIlraith and Fadel, 2002)
 - Serviços Web (SW)
 - Web Semântica
- Serviços Web Semânticos (SWS)

Serviços Web Semânticos

- Web Semântica na descrição e no uso de SW
- Interoperabilidade: Serviços Web
- Interoperabilidade dinâmica: Serviços Web Semânticos
 - Automatizar
 - Descoberta
 - Invocação
 - Composição e
 - Interoperabilidade entre os serviços

OWL-S

- Ontologia para serviços expressa em OWL
 - Classes, hierarquia de classes, relacionamentos, etc.
 - OWL DL
 - Descrever funcionalidades e publicação de serviços
- Principal foco: descoberta e composição
- Inclui três principais sub-ontologias (Martin et al., 2007)
 - *Service profile*: descreve “o que o serviço faz”
 - *Process model*: descreve “como o serviço é utilizado”
 - *Grounding*: descreve “como interagir com o serviço”

OWL-S: IOPR

- IOPR: *Inputs, Outputs Preconditions e Results*
 - *Inputs*: descrição dos objetos que o serviço consome e processa
 - *Outputs*: descrição dos objetos que o serviço produz
 - *Preconditions*: uma proposição que deve ser verdadeira para o serviço operar eficientemente
 - *Results*: consiste de um efeito (*Effects*) e especificações de *Output*
 - *Effect*: uma proposição que vai se tornar verdadeira quando o serviço terminar sua execução

OWL-S: *Profile*

- Provê um conjunto de conceitos para especificar as funcionalidades dos serviços
- Suporta a descoberta de serviços
 - Baseada nos IOPRs
 - Provedores de serviço publicam o *Profile* dos seus serviços
 - Cliente de serviço especificam as funcionalidades do serviço esperado
 - *Semantic matching* entre as duas especificações

Motivação

- *Semantic Matching* pode
 1. descobrir um ou mais serviços
 2. não descobrir serviço algum
 3. descobrir vários sub-serviços parciais
 - Oportunidade para composição de serviços

Motivação

- Composição automática é o processo de automaticamente **planejar**, **selecionar** e **executar** serviços Web para alcançar o objetivo do usuário (Ankolekar et al., 2003)
 - “Fazer reservas de viagem para a Conferência Web W3C Brasil 2009”
 - Onde ocorrerá evento?
 - Quais as datas de início e fim?
 - Que tipo de inscrição eu vou pagar?
 - Que tipo de viagem – navio, ônibus, avião?
 - Quais companhias fazem a rota e têm disponibilidade de assento?
 - Há necessidade de traslado?
 - Há hotéis com disponibilidade?

Composição de SWS

- Planejamento
 - Descobrir sub-serviços parciais
 - Montar *workflow*
- Seleção
 - Otimizar a composição
 - Selecionar os melhores serviços
 - Atributos funcionais ou não funcionais
- Invocação (ou execução)
 - Garantir a execução da composição efetuada nas fases de descoberta, planejamento e seleção de serviços

Composição de SWS

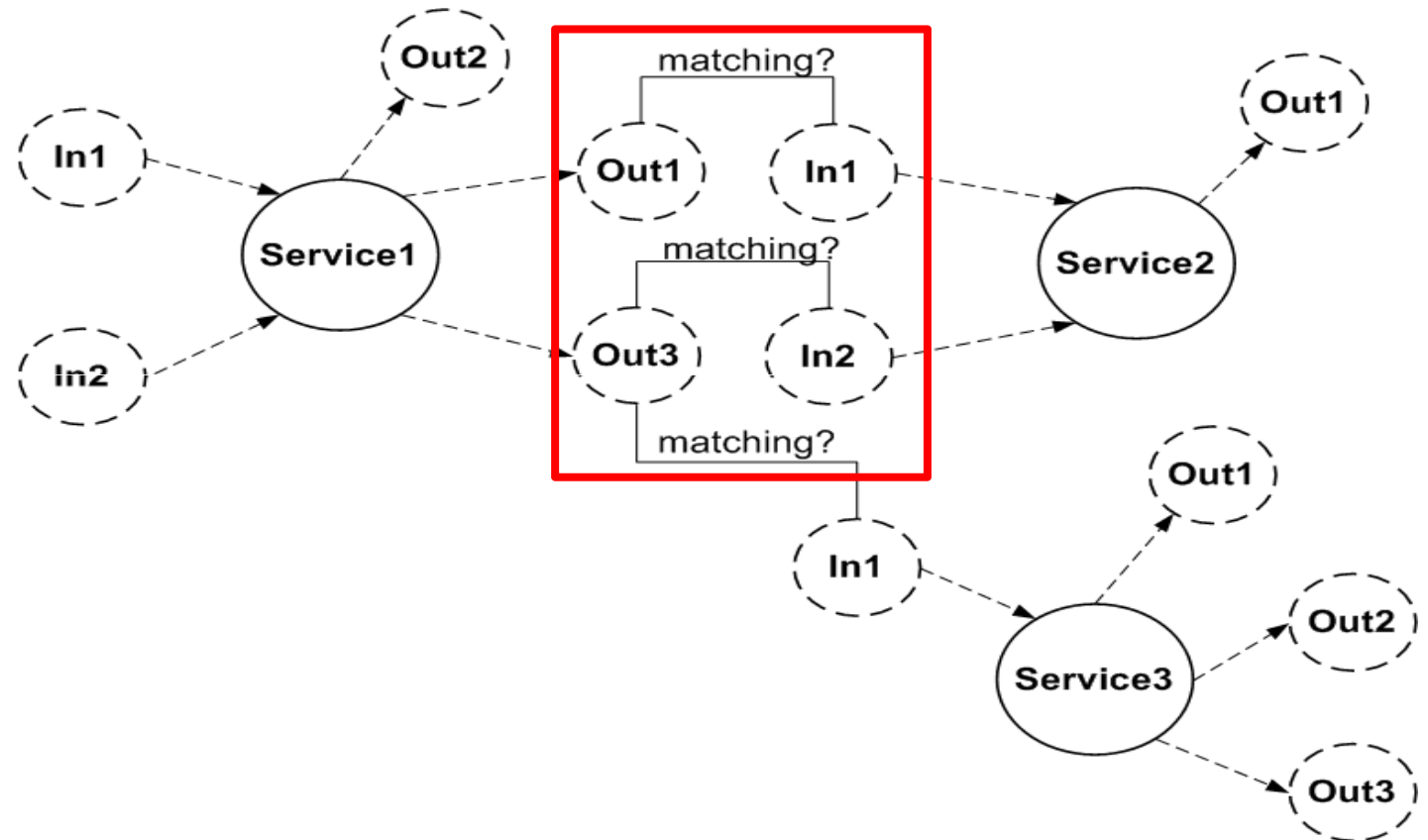
- Planejamento e seleção
 - Envolvendo descoberta
 - Atributos funcionais e não-funcionais
 - Planejamento e seleção estão fundidas em uma única etapa
 - Este trabalho não trata questões de invocação automática

Composição de SWS

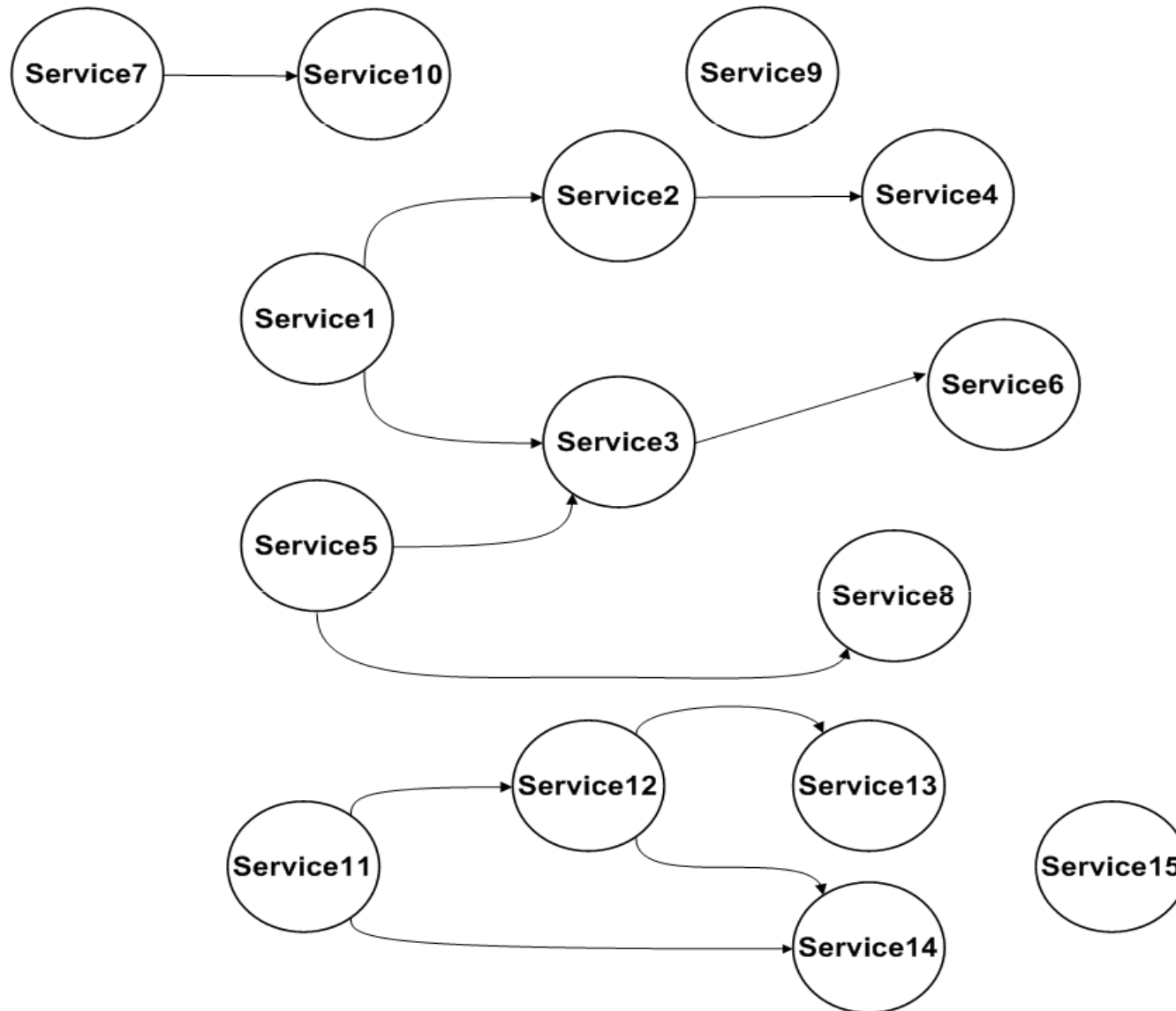
- Modelagem em grafos direcionados e com custos estabelecidos
- Grafo criado na publicação do serviço
 - Reduz o tempo de composição
- Políticas de custos e descontos
- Algoritmo de Dijkstra (Bellman-Ford)
- Grafo da composição final
- Descrição da composição final em OWL-S
- Verificação da composição final

Modelo em grafo para SWS

- Um par de serviços são conectados, com aresta de saída, apenas uma vez
 - Pode haver mais de um par entrada/saída correspondentes entre si



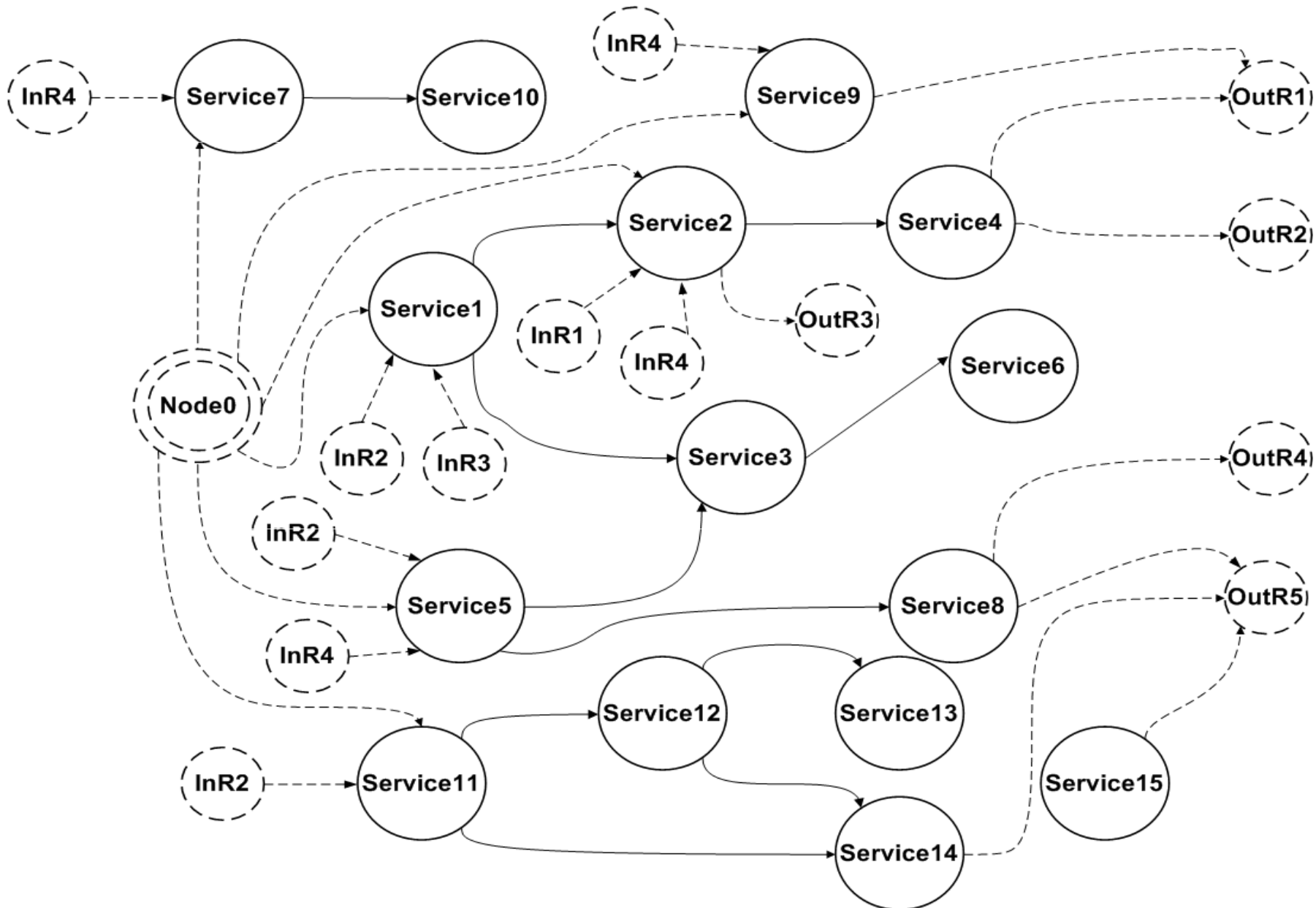
Modelo em grafo para SWS



Modelo em grafo para SWS

- Nós artificiais
 - Nó raiz único
 - Conectado a todos os serviços que possuem ao menos uma entrada disponível
 - Diminui o espaço de busca do algoritmo de cálculo do caminho de custo mínimo
 - Nó para cada saída requerida
 - Cada serviço que contem a saída requerida representada pelo nó artificial é conectado ao nó
 - Garante que todas as saídas requeridas, se existirem no grafo, estarão na composição final

Modelo em grafo para SWS



Política de custo

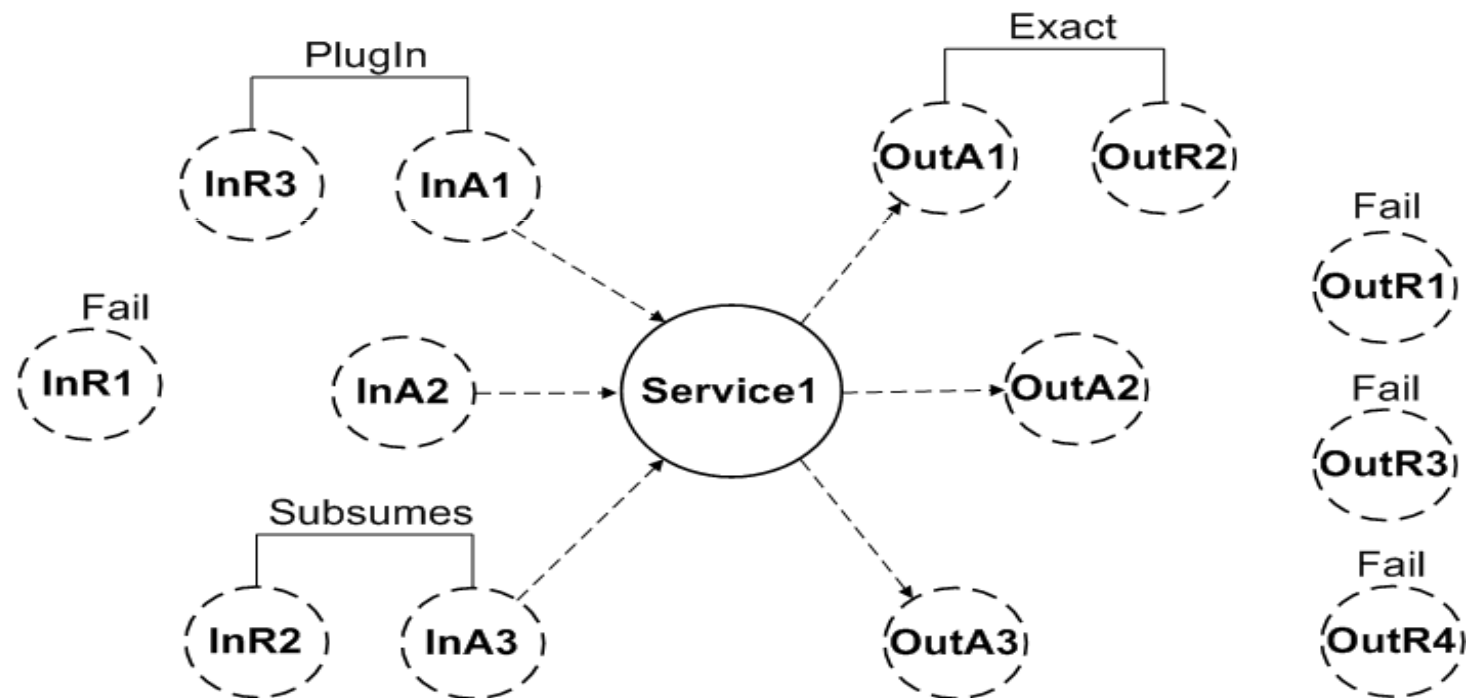
- Custo baseado na funcionalidade
 - Utiliza *Semantic Matching*
- Custo máximo é atribuído ao serviço no momento de sua publicação
 - Quando o serviço é adicionado ao grafo
- Estratégia
 - Assume o custo máximo e oferece descontos

Custo inicial do serviço

- $C_{is} = (N_{in} * 3) + (N_{out} * 3)$
 - C_{is} : custo inicial do serviço
 - N_{in} : número de entradas do serviço que é multiplicado pelo grau 3 (pior caso)
 - N_{out} : número de saídas do serviço que é multiplicado pelo grau 3 (pior caso)
- Alguns serviços devem ter custos diferenciados
 - Determinante do comportamento do algoritmo
 - Descontos

Descontos funcionais

- Serviços que mais se aproximem da requisição inicial do usuário
 - Entradas disponíveis e saídas requeridas → desconto
- Algoritmo de custos mínimos
 - Prioriza entradas disponíveis e/ou saídas requeridas



Desconto para as entradas

$$DT_{in} = \left(\sum_{i=1}^{N_{in}} D_{in} \right) * \frac{1}{\lambda}, \text{ em que:}$$

- DT_{in} : desconto total em relação às entradas do serviço;
- N_{in} : número de entradas do serviço;
- D_{in} : desconto de cada entrada do serviço
- λ : é um parâmetro ajustável pelo usuário. O valor padrão de λ é 1.

Matching	Ocorrência	Desconto
<i>Exact</i>	Se InR e InServico são equivalentes	3
<i>PlugIn</i>	Se InR é sub-conceito de InServico	2
<i>SubSume</i>	Se InR é super-conceito de InServico	1
<i>Fail</i>	InR e InServico não possuem similaridade semântica	0

Desconto para as saídas

$$DT_{out} = \left(\sum_{i=1}^{N_{out}} D_{out} \right) * \frac{1}{\lambda}, \text{ em que:}$$

- DT_{out} : desconto total em relação às saídas do serviço;
- N_{out} : número de saídas do serviço;
- D_{out} : desconto de cada saída do serviço
- λ : é um parâmetro ajustável pelo usuário. O valor padrão de λ é 1.

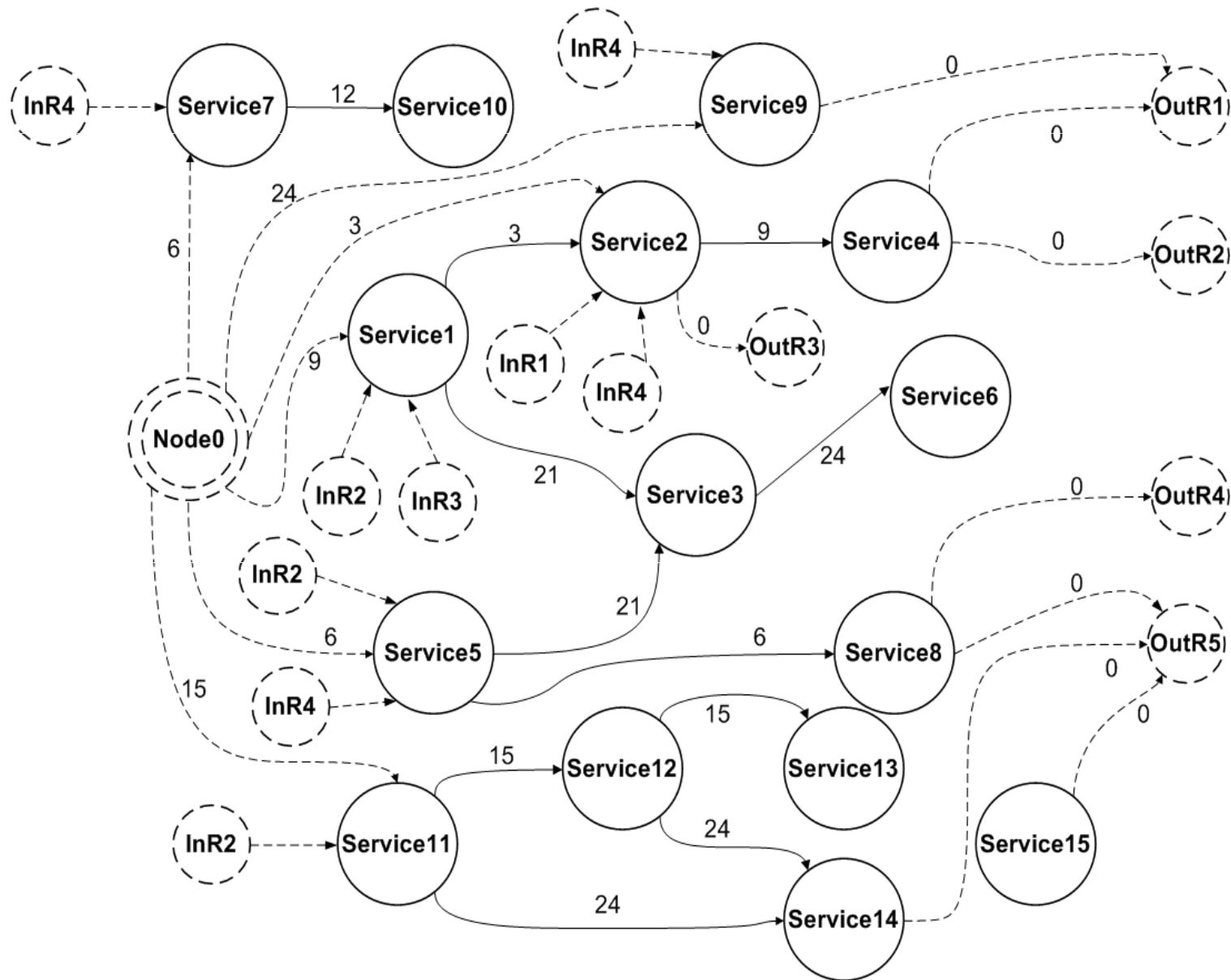
Matching	Ocorrência	Desconto
<i>Exact</i>	Se OutR e OutServico são equivalentes	3
<i>PlugIn</i>	Se OutR é sub-conceito de OutServico	2
<i>SubSume</i>	Se OutR é super-conceito de OutServico	1
<i>Fail</i>	OutR e OutServico não possuem similaridade semântica	0

Custo efetivo do serviço

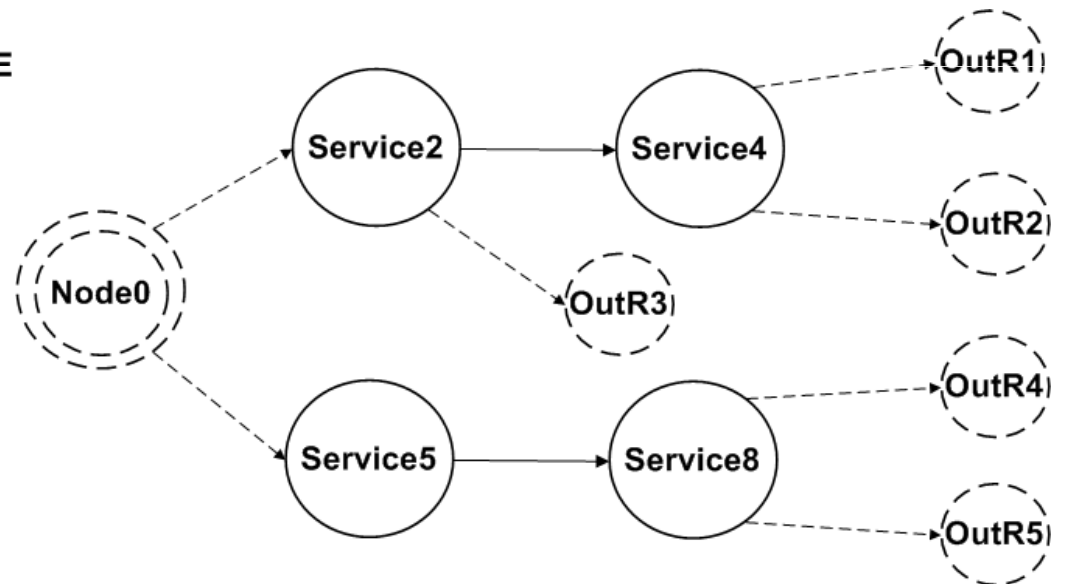
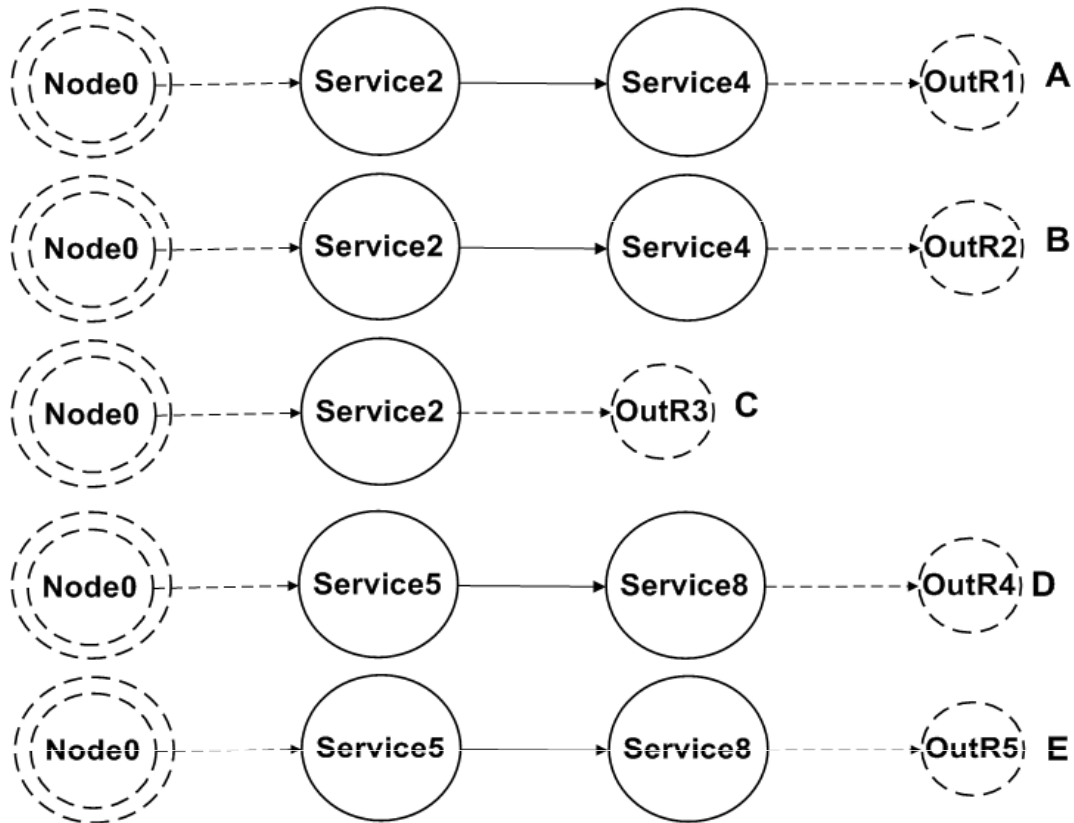
- $Ces = Cis - DTs$
 - Ces: custo efetivo do serviço
 - Cis: custo inicial do serviço
 - DTs: desconto total do serviço

- $DTs = DTin + DTout$
 - DTs: desconto total do serviço
 - DTin: desconto total em relação às entradas do serviço
 - DTout: desconto total em relação às saídas do serviço

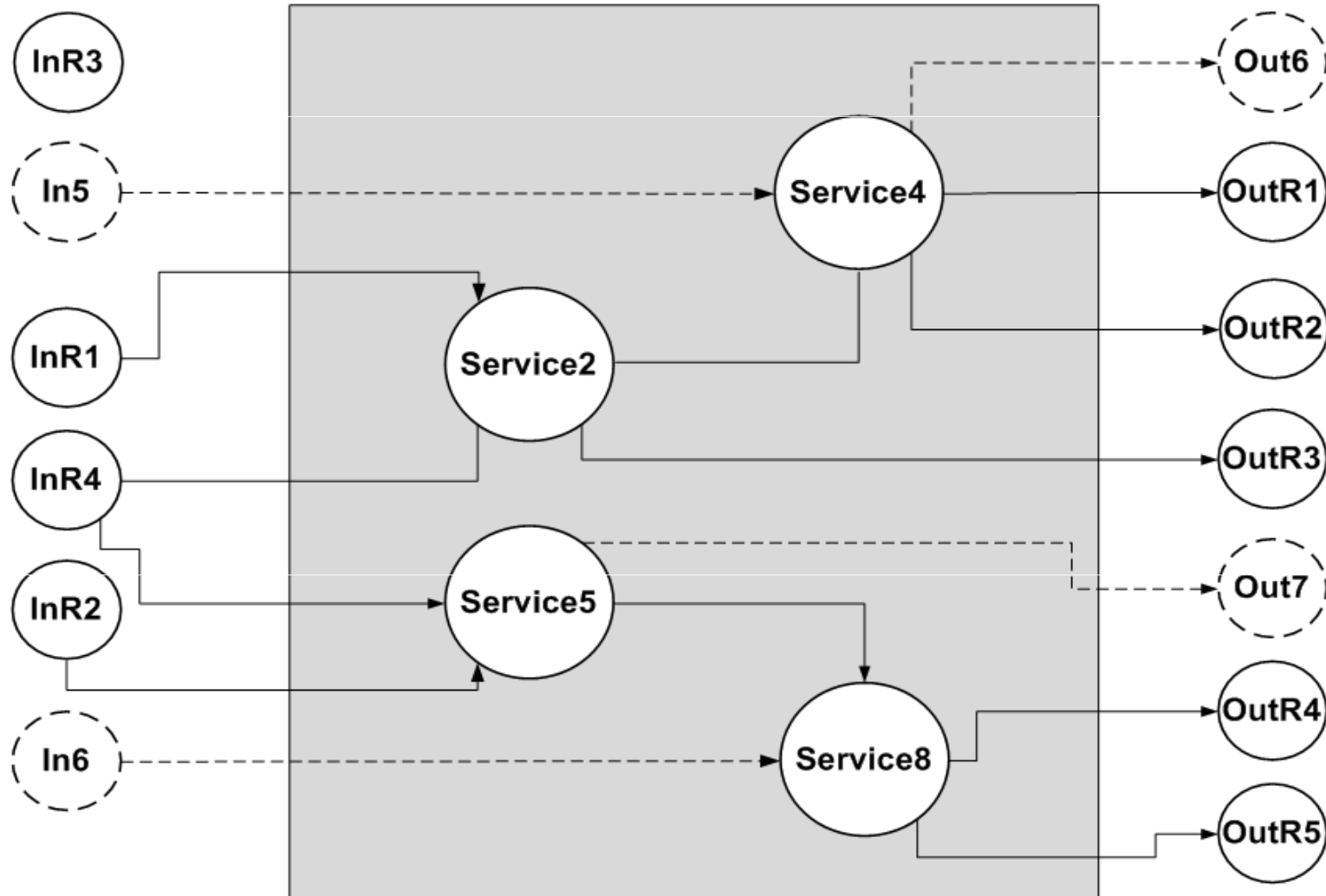
Grafo com custos e descontos funcionais



Caminhos mínimos & Serviço composto



Verificação da composição final



Mapeamento da composição final para o OWL-S

Grafo	Construtor OWL-S
Caminho seqüencial	<i>Sequence</i>
Caminho paralelo com barreira de sincronização apenas no início (<i>AND-Split</i>)	<i>Split</i>
Caminho paralelo Caminho paralelo com barreiras de sincronização no início e no final (<i>AND-Split+AND-Join</i>)	<i>Split+Join</i>
Caminho paralelo sem barreiras de sincronização	Dois construtores <i>Sequence</i>
<i>OR-Split</i> com exatamente duas opções	<i>If-Then-Else</i>
<i>OR-Split</i> com mais de duas opções	<i>Choice</i>
<i>Iteration</i>	<i>Repeat-While</i> ou <i>Repeat-Until</i>

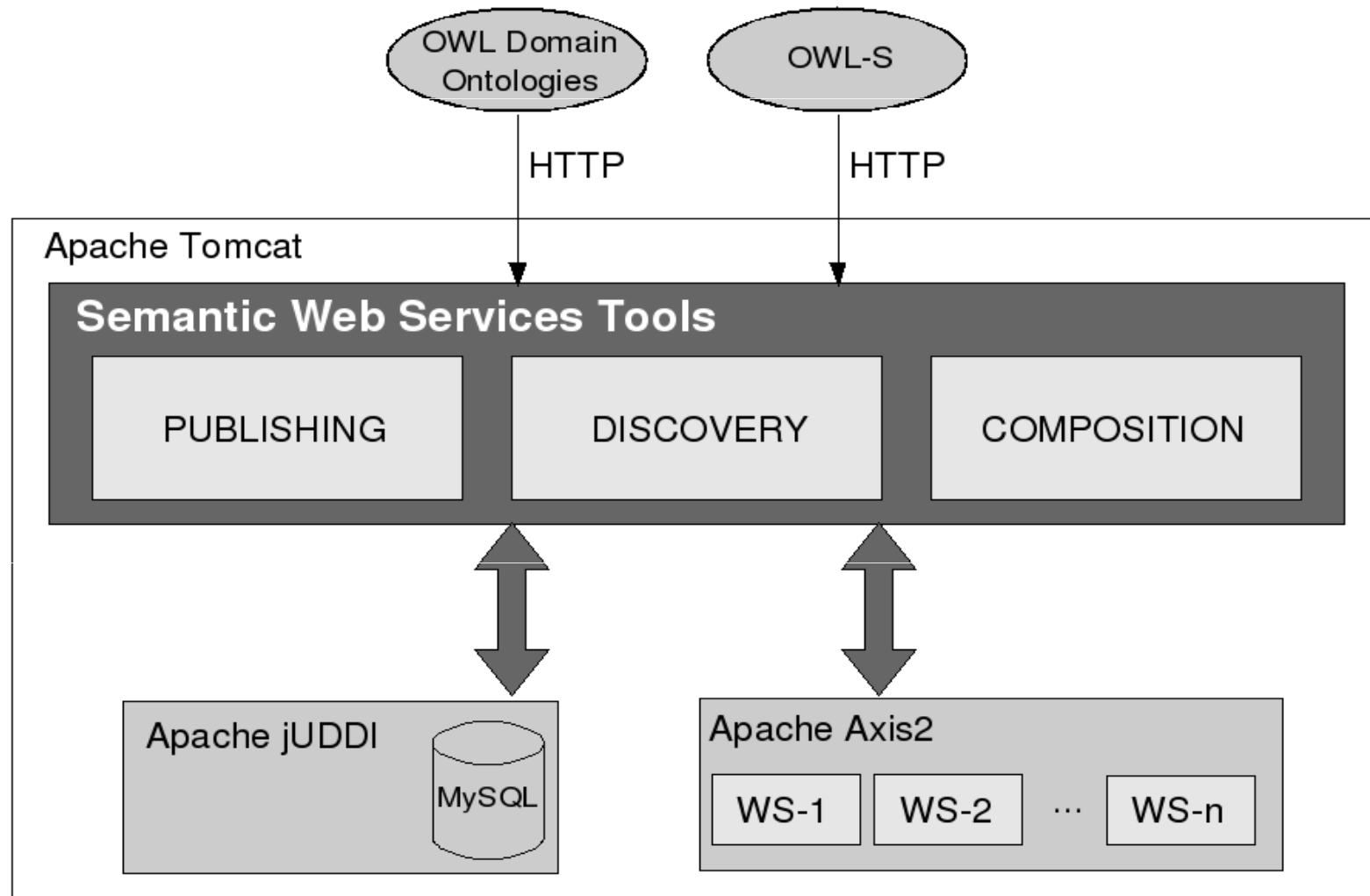
Trabalhos relacionados

- Técnicas de planejamento em Inteligência Artificial
 - Golog, preferências e restrições (McIlraith and Son, 2002;)
 - HTN, seqüencia de ações sub-tarefas (Paolucci et al., 2003a; Sirin et al., 2004b; Kuter et al., 2004)
- Técnicas para síntese de programa
 - Síntese Estrutural de Programas (SSP) (Lammermann, 2002)
 - Lógica Linear (Rao, 2004)

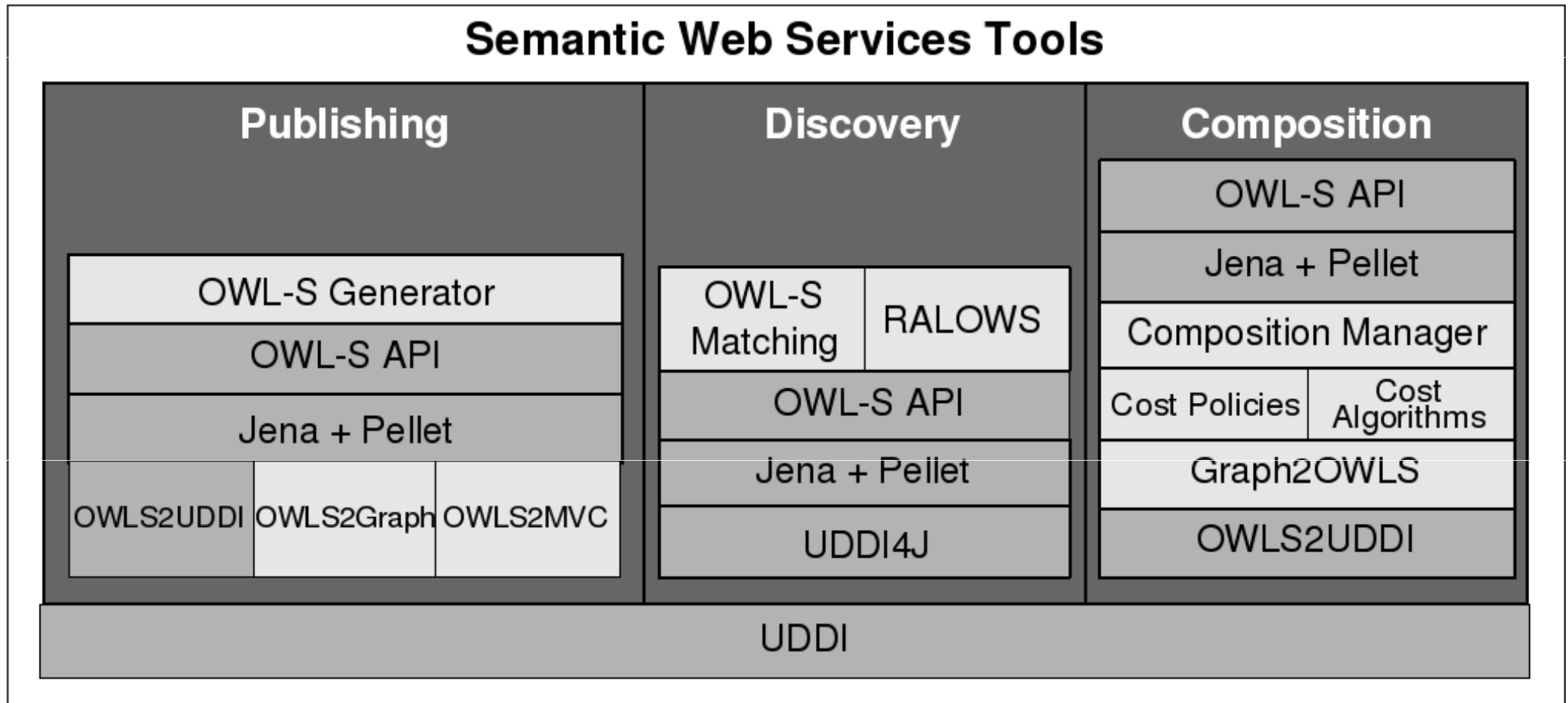
Trabalhos relacionados

- Técnicas baseadas em *workflow*
 - Autômato, dependência ES, sem custos, entradas não disponíveis (Hashemian and Mavaddat, 2005; Mitra et al., 2007)
 - Grafos e similaridade semântica
 - Sem custos, caminho inverso, sem seleção do melhor serviço (muitas composições), todas possibilidades (Silva et al., 2007)
 - Custo (tempo e similaridade), caminho mínimo de cada entrada para cada saída, não trata intersecções (Arpinar et al., 2005)
 - Custo (não define), estende Dijkstra (Cui et al., 2005)
 - Nenhum deles utilizam custos funcionais, nós artificiais e descontos

Infraestrutura para SWS



Arquitetura da implementação



Cinza claro:
 Cinza médio:
 Cinza escuro:

Avaliação da infraestrutura implementada

- Configuração dos experimentos executados
- Tempo de resposta: descoberta de serviço no repositório UDDI
- Tempo de resposta para a publicação de serviço

Varriável	Varição
Número de serviços	10 – 1000
Número de parâmetros por serviços	10 – 20

Número de serviços	Tempo de descoberta (ms)
10	879
100	947
500	1076
1000	1126

Função	Complexidade	Tempo (ms)
Registro no UDDI	–	5480
Descoberta $Out \rightarrow In$	–	1126
Descoberta $In \rightarrow Out$	–	1126
Conexões $Out \rightarrow In$	$O(N_{Out})$	< 10
Conexões $In \rightarrow Out$	$O(N_{In})$	< 10
Tempo total	–	< 7752

Avaliação da infraestrutura implementada

- Tempo de resposta para a composição de serviço

Função	Complexidade	Tempo (ms)
Descoberta OutR	–	1126
Descoberta InR	–	1126
Conectar <i>Node0</i>	$O(N_s)$	< 10
Conectar saídas requeridas	$O(N_s * N_{OutR})$	< 10
Calculo dos descontos para entradas	$O(N_s * N_{InR})$	< 10
Calculo dos descontos para saídas	$O(N_s * N_{OutR})$	< 10
Gerar grafo transposto	$O(N_s * N_{adj})$	< 60
Aplicar descontos	$O(N_s * N_{adj})$	< 60
Gerar grafo transposto (volta)	$O(N_s * N_{adj})$	< 60
Dijkstra por saída	$O(N^2)$	< 10
Tempo total	–	< 2482

Trabalhos futuros

- Algumas direções para trabalhos futuros
 - Avaliar a utilização de outros custos (não funcionais)
 - Ex: QoS, preço (\$\$\$)...
 - Customização da composição final a partir de custos não funcionais
 - Escolha de composições alternativas
 - Invocação automática das composições finais
 - Mapeamento para WS-BPEL
 - Apache ODE



Descoberta e composição de Serviços Web Semânticos: de condições a alternativas

Cássio Prazeres (prazer@unifacs.br) - UNIFACS

Cesar Augusto Camillo Teixeira (cesar@dc.ufscar.br) – UFSCar

Maria da Graça Pimentel (mgp@icmc.usp.br) – USP

Conferência Web W3C Brasil'09: 24/Novembro/2009