Capturing Web services Provider Constraints – An Algorithmic Approach

Sudeep Mallick & S. V. Subrahmanya
Infosys Technologies Ltd.
Bangalore, India.
({sudeepm, subrahmanyasv}@infosys.com)

CBSE 2007, Medford, MA
July 10, 2007

Agenda

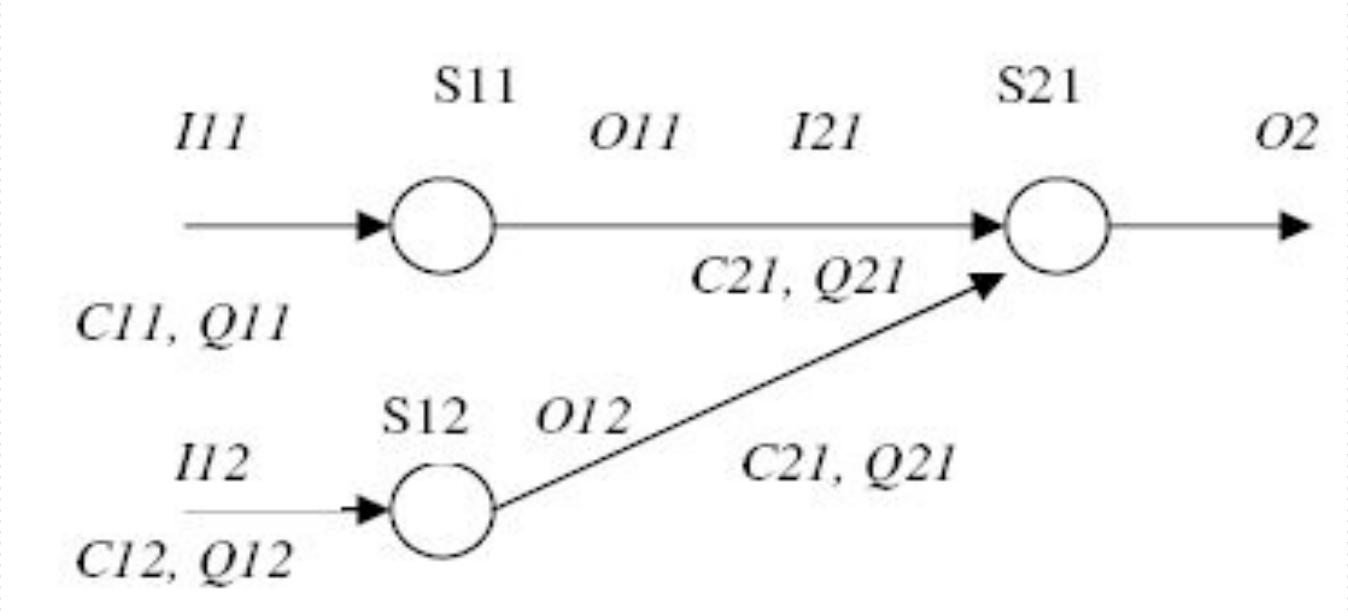
- Context
- Understanding Services and Web services
- Special Service Provider configurations
- Proposed algorithm
- Results
- Implementation – proposed tool
- Conclusion

Service provisioning

- Multiple service providers
- Services available at multiple cost and quality combinations
- Gold, silver, bronze, etc…
- Complex service provider implementation combinations and dependencies

Service provider coalitions

- Service providers coalitions and competitions



The problem

- Optimal selection of service implementations (concrete services) from among various service providers
  - Cost
  - Quality of Service
- Exogenous and endogenous cases

Criticality of the problem

- Garnter: SOA will be used in more than 50% of new, mission critical applications designed in 2007 and more than 80% by 2010 with 0.7 probability
- www.amazon.com
- www.google.com
- www.strikeiron.com
- www.esri.com/software/arcwebservices/
- www.globexplorer.com
- SaaS – Software as a Service model

Service provider configurations - 1

- Same service available from the same service provider at different price / QoS attribute value

Service	Provider	Cost of operation (or price)	QoS attribute (reliability)
1 (S11)	1	$1	0.75
1 (S11')	1	$1.1	0.80
1 (S11'')	1	$1.2	0.90
2 (S21)	1	50 cents	0.75
2 (S21')	1	$1	0.90

Service provider configurations - 2

- Same service could be available at different cost / QoS attribute value from the same service provider when bundled (composite services) with other services from the same provider, in case the interfaces match

Service	Provider	Cost of operation (or price)	QoS attribute (response time)
1 (S11)	1	$1	90 ms
2 (S21)	1	$1	80 ms
1-2 (composite – S11C, S21C)	1	$1.75	160 ms

Service provider configurations - 3

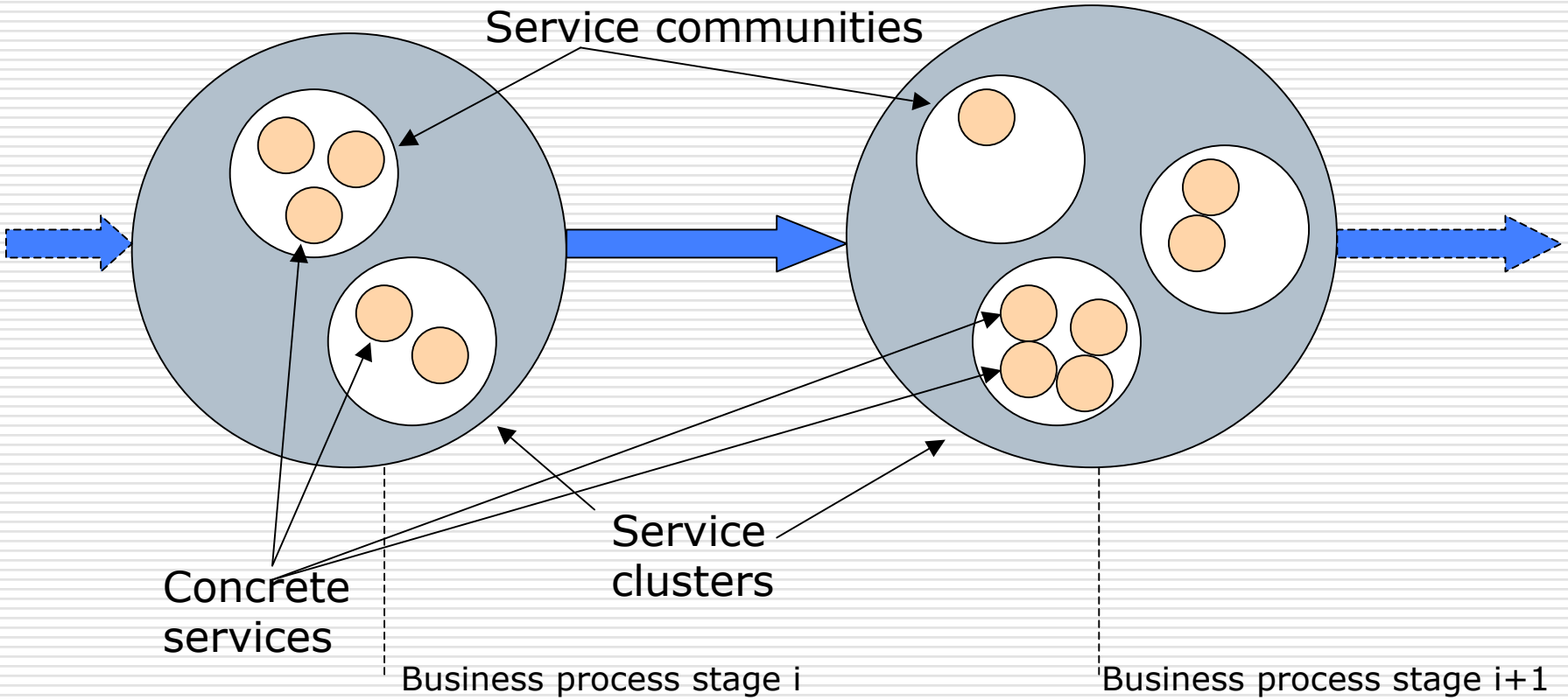
- Two or more service providers could enter into contractual agreements for preferential deals when operating as a coalition

Service	Provider	Cost of operation (or price)	QoS attribute (response time)
1 (S11)	1	$1	90 ms
2 (S21)	1	$1	80 ms
3 (S32)	2	$1.5	70 ms
4 (S42)	2	$1.25	85 ms
5 (S52)	2	$1	70ms
1-2-3-4-5 (composite – S11C', S21C', S32C', S42C', S52C')	1,2	$5.40	370 ms
1-2 (composite – S11C, S21C)	1	$1.75	160 ms
3-4 (composite – S32C, S42C)	2	$2.70	150 ms

Formal representation of services

- Service enabled business process as a weighted multi-stage graph
- Weights representing cost / quality
- Service clusters: services grouped into business process stages representing same functionality
- Service communities: matching input and output interfaces

Clusters and communities



Definitions

- Web service s is a tuple *s(p, f, i, o, c, q, m)*
  - *p = provider; f = functionality; i = input interface; o = output interface; c = cost; q = aggregated quality metric; m = composite number*
- *Web services cluster CL is a set of concrete services that provide functionality F*
  - *CL(F) = {s | s.f = F}*
- *Web services community CM is a set of concrete services in a cluster that have the same interfaces I, O*
  - *CL(F, I, O) = {s | s E CL(F) Λ s.i = I Λ s.o = O}*

Above formalizations are built upon the basics presented in Gao, Y., Na, J., Zhang, B., Yang, L., Gong, Q.: Optimal Web Services Selection Using Dynamic Programming, Proceedings of the 11th IEEE Symposium on Computers and Communications (ISCC'06) (2006)

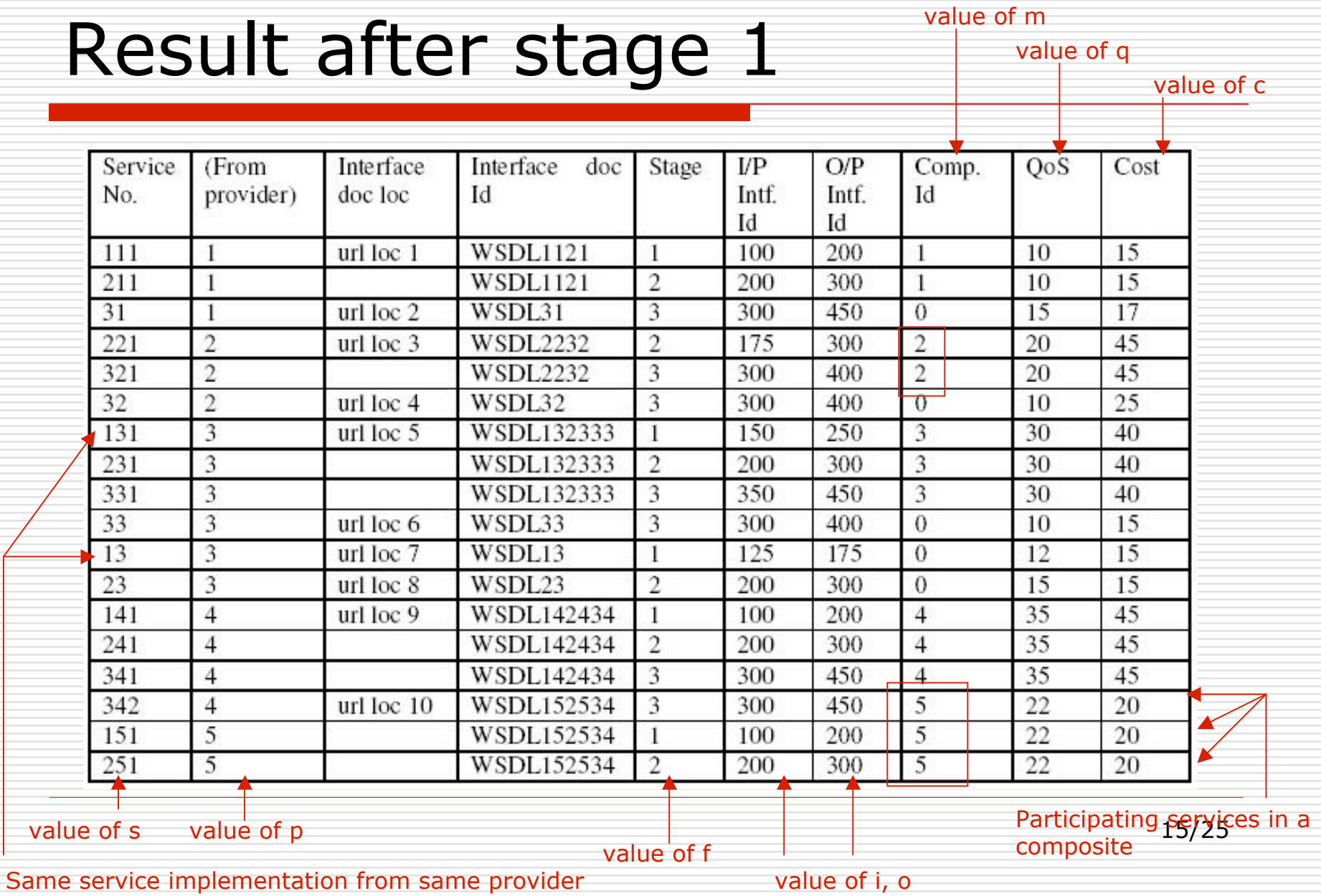
Proposed algorithm – Stage 1

- Define the service tuple *S(s,p,f,i,o,c,q,m)* and create service description data structure
- Create uniquely identifiable services for multiple cost-quality combinations from same provider
- Set value of *m* for composites

Input from service providers

Sl.No	(From Provider)	Interface doc loc	Stage	Composite	QoS	Cost
1	1	url loc 1	1	Yes with 1	10	15
2	1		2	Yes with 1	10	15
3	1	url loc 2	3	0	15	17
4	2	url loc 3	2	Yes with 2	20	45
5	2		3	Yes with 2	20	45
6	2	url loc 4	3	0	10	25
7	3	url loc 5	1	Yes with 3	30	40
8	3		2	Yes with 3	30	40
9	3		3	Yes with 3	30	40
10	3	url loc 6	3	0	10	15
11	3	url loc 7	1	0	12	15
12	3	url loc 8	2	0	15	15
13	4	url loc 9	1	Yes with 4	35	45
14	4		2	Yes with 4	35	45
15	4		3	Yes with 4	35	45
16	4	url loc 10	3	Yes with 5	22	20
17	5		1	Yes with 5	22	20
18	5		2	Yes with 5	22	20

Result after stage 1



Service No.	(From provider)	Interface doc loc	Interface doc Id	Stage	I/P Intf. Id	O/P Intf. Id	Comp. Id	QoS	Cost
111	1	url loc 1	WSDL1121	1	100	200	1	10	15
211	1		WSDL1121	2	200	300	1	10	15
31	1	url loc 2	WSDL31	3	300	450	0	15	17
221	2	url loc 3	WSDL2232	2	175	300	2	20	45
321	2		WSDL2232	3	300	400	2	20	45
32	2	url loc 4	WSDL32	3	300	400	0	10	25
131	3	url loc 5	WSDL132333	1	150	250	3	30	40
231	3		WSDL132333	2	200	300	3	30	40
331	3		WSDL132333	3	350	450	3	30	40
33	3	url loc 6	WSDL33	3	300	400	0	10	15
13	3	url loc 7	WSDL13	1	125	175	0	12	15
23	3	url loc 8	WSDL23	2	200	300	0	15	15
141	4	url loc 9	WSDL142434	1	100	200	4	35	45
241	4		WSDL142434	2	200	300	4	35	45
341	4		WSDL142434	3	300	450	4	35	45
342	4	url loc 10	WSDL152534	3	300	450	5	22	20
151	5		WSDL152534	1	100	200	5	22	20
251	5		WSDL152534	2	200	300	5	22	20

Proposed algorithm – Stage 2

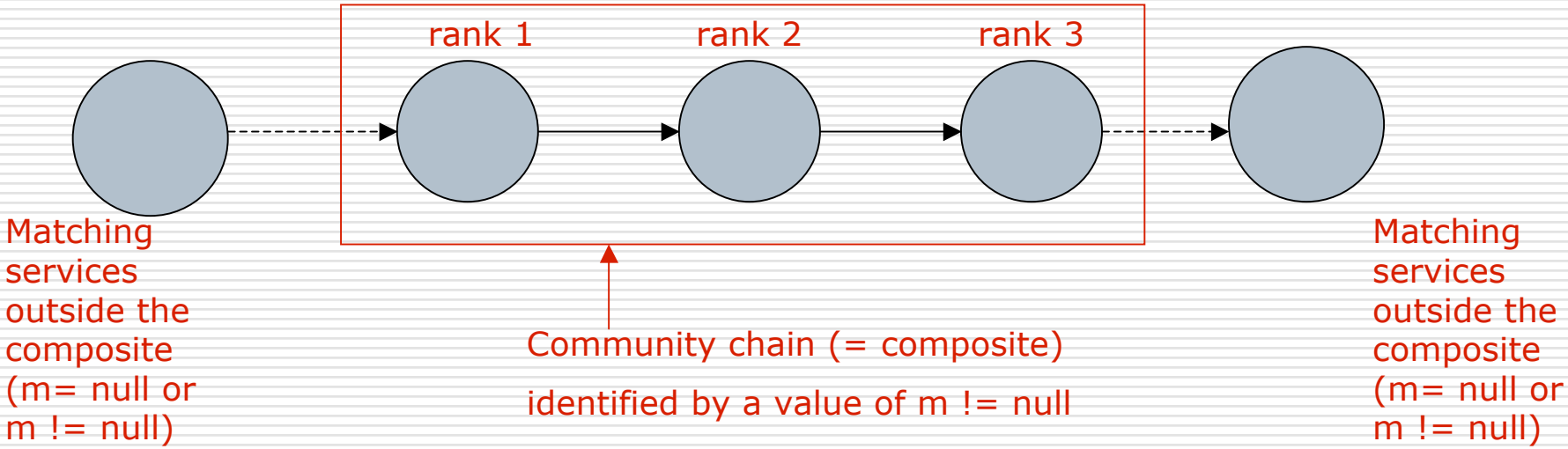
- Partition initial data into clusters (same value for *f*)
- Partition each resulting cluster into data groups having same values for *i* and *o* and *m* = *null*
- Partition each resulting cluster into data groups having same values for *i* and *o* and *m* != *null*
- For communities having *m* !=*null* create a link in a community chain data structure (CC) having same value of *m* and create rank *k*

Proposed algorithm – Stage 3

- Pre-condition
  - List of clusters and communities ready from stage 2
- Create matching among communities in cluster *j* and *j+1* for all clusters
  - Cluster *j*: IF (community NOT IN CC) OR ((community IN CC) AND (is the last position community for an *m*))
  - Cluster *j+1*: (community NOT IN CC) OR ((community IN CC) AND (is the last position community for an *m*))
    - THEN match the communities
      - IF matching THEN create matches
- For communities in CC
  - For each value of m establish matches
    - From *k = 2* to largest value of rank *k*
      - Create matches between communities having rank *k-1* and *k*

Stage 3

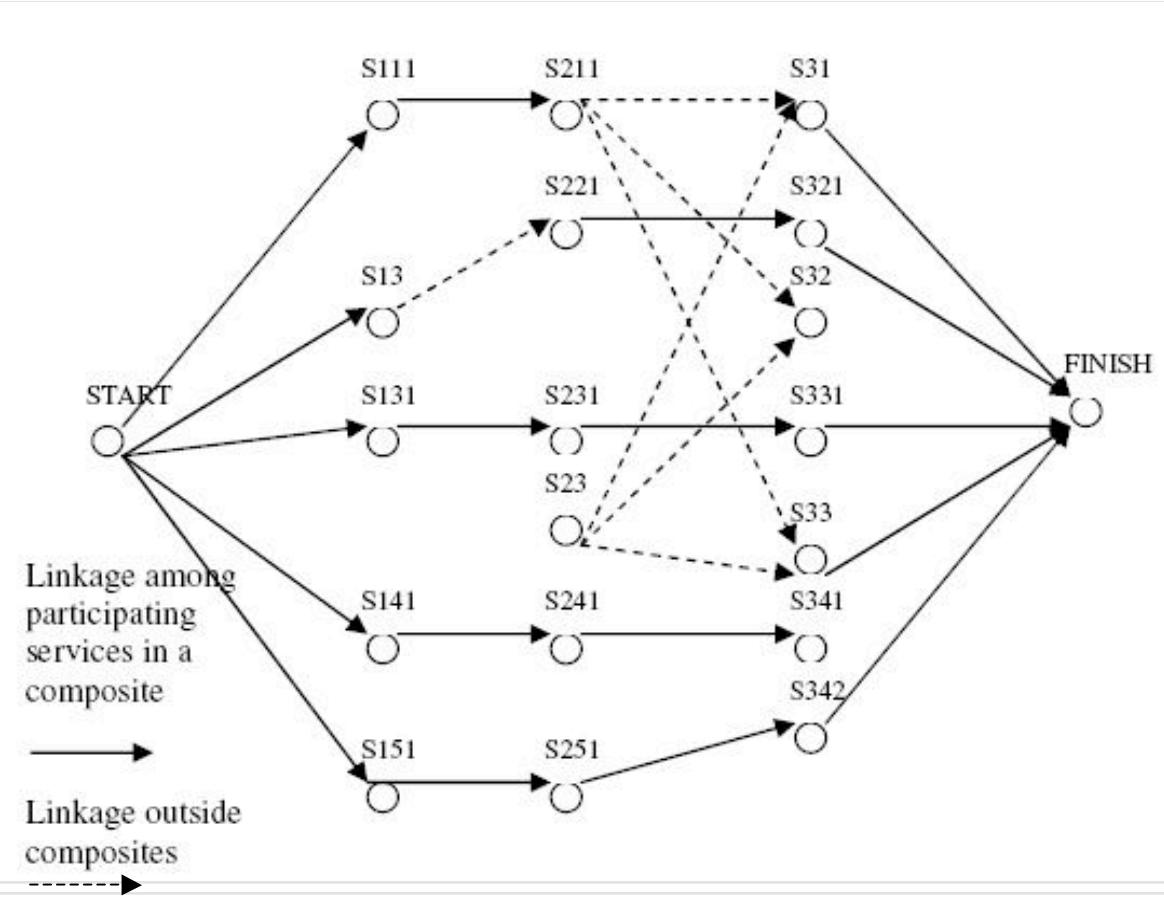
- Post condition
  - Set of matched communities and services contained therein



Results

Service No.	(From provider)	Interface doc loc	Interface doc Id	Stage	I/P Intf. Id	O/P Intf. Id	Comp. Id	QoS	Cost
111	1	url loc 1	WSDL1121	1	100	200	1	10	15
211	1		WSDL1121	2	200	300	1	10	15
31	1	url loc 2	WSDL31	3	300	450	0	15	17
221	2	url loc 3	WSDL2232	2	175	300	2	20	45
321	2		WSDL2232	3	300	400	2	20	45
32	2	url loc 4	WSDL32	3	300	400	0	10	25
131	3	url loc 5	WSDL132333	1	150	250	3	30	40
231	3		WSDL132333	2	200	300	3	30	40
331	3		WSDL132333	3	350	450	3	30	40
33	3	url loc 6	WSDL33	3	300	400	0	10	15
13	3	url loc 7	WSDL13	1	125	175	0	12	15
23	3	url loc 8	WSDL23	2	200	300	0	15	15
141	4	url loc 9	WSDL142434	1	100	200	4	35	45
241	4		WSDL142434	2	200	300	4	35	45
341	4		WSDL142434	3	300	450	4	35	45
342	4	url loc 10	WSDL152534	3	300	450	5	22	20
151	5		WSDL152534	1	100	200	5	22	20
251	5		WSDL152534	2	200	300	5	22	20

Results


S111
S211
S31
S221
S321
S13
S32
START
FINISH
S131
S231
S331
S23
S33
Linkage among participating services in a composite
S141
S241
S341
S342
S151
S251
Linkage outside composites

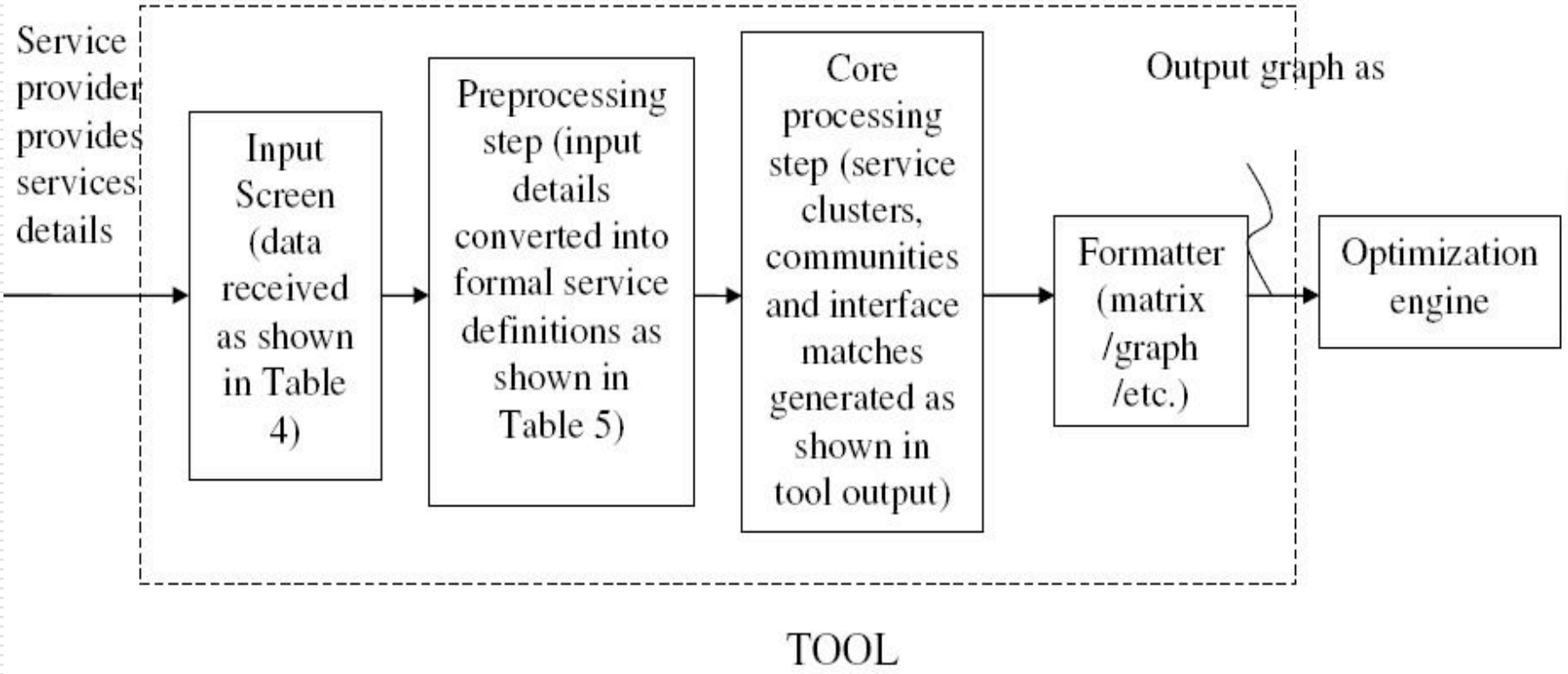
Generality of the algorithm

- Single provider, multiple simple services at single or variable cost – quality combinations
- Single provider, multiple composite services at single or variable cost – quality combinations
- Multiple provider, multiple services (simple / composite) at single or variable cost – quality combinations with composites formed from the same respective service providers
- Multiple provider, multiple services (simple / composite) at simple or variable cost – quality combinations with coalitional composites formed across multiple service providers
- Combination of simple and composite services at various cost – quality combinations for all the above conditions

Analysis of the algorithm

- Stage 1 – O(N): N is total number of services
- Stage 2 – O(N)
- Stage 3 – O((K-1)(N/K)^2): K is total number of functional stages i.e. constant for a problem => O(N^2)
- Hence, the proposed algorithm is O(n^2) primarily due to the matching stage

Proposed tool



Conclusion

- Formal representation of varying cost-quality combinations
- Formal representation of coalitional constraints among service providers
- Algorithm for derivation of DAG from the formal representation
- Resulting DAG could be subjected to optimization methods (IP, DP, etc.)

- Questions