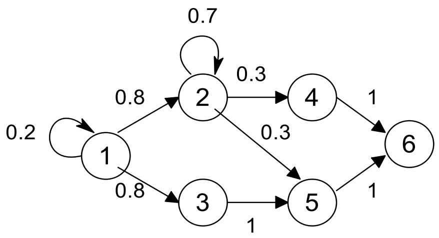
Appendix 2: Sample examples

Example A-1: Composition graph summarization without conditional pattern for a SC (service composition) which is the first member of Comm population

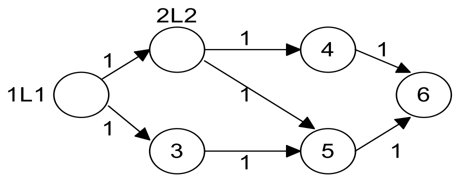
Composition graph



Composition graph with 6 functions

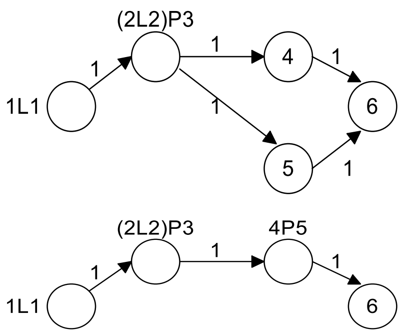
SC_1 ={341,797,890,1635,1936,2130}
SC_1.Q={(0.7900,0.5684,0.0190), (0.8800,0.0349,0.0046), (0.8500,0.0249,0.0027), (0.9100,0.0782,0.0021), (1,0.0156,2.4153e-04), (0.9600,0.0205,0.0103)}

#web service	Q=(readiness, response-time, cost)
1	(0.7900,0.5684,0.0190)
2	(0.8800,0.0349,0.0046)
3	(0.8500,0.0249,0.0027)
4	(0.9100,0.0782,0.0021)
5	(1,0.0156,2.4153e-04)
6	(0.9600,0.0205,0.0103)



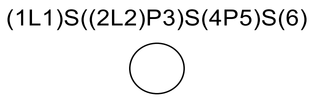
First step: removing the loop pattern (services 1 and 2)

#web service	Q=(readiness, response-time, cost)
1L1	(0.7506,0.7105,0.0238)
2L2	(0.6875,0.1164,0.0152)
3	(0.8500,0.0249,0.0027)
4	(0.9100,0.0782,0.0021)
5	(1,0.0156,2.4153e-04)
6	(0.9600,0.0205,0.0103)



Second step: removing the parallel pattern

#web service	Q=(readiness,response-time,cost)
1L1	(0.7506,0.7105,0.0238)
(2L2)P3	(0.5844,0.1164,0.0178)
4P5	(0.9100,0.0782,0.0023)
6	(0.9600,0.0205,0.0103)



Third step: removing the sequence pattern and generating Summary service

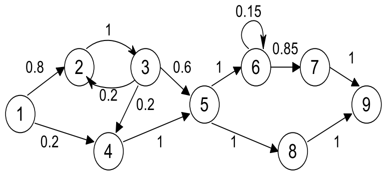
# web service	Q=(readiness,response-time,cost)
(1L1)S((2L2)P3)S(4P5)S6	(0.3832,0.9256,0.0542)

Q'= (0.3832,0.9256,0.0542)
Comm=(SC,Q')$_1$

Example A-2: Composition graph summarization with conditional pattern for a SC (service composition) which is the first member of Comm population

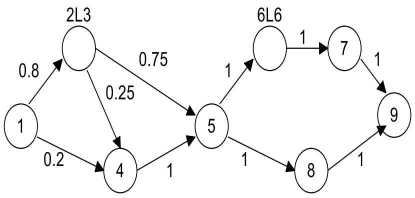
Composition graph

SC_1={10,401,664,1049,1336,1445,1808,2074,2409}



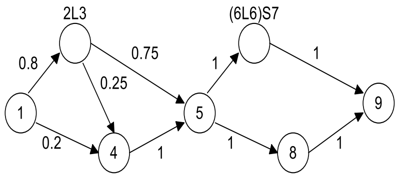
Composition graph with 9 functions

#web service	Q=(readiness,response-time,cost)
1	(0.9600,0.0187,0.0101)
2	(0.7200,0.2384,4.8305e-04)
3	(0.9300,0.0121,0.0022)
4	(1,0.0392,0.0062)
5	(0.8800,0.1705,0.1151)
6	(0.8300,0.0154,7.2458e-04)
7	(0.9600,0.0397,0.0121)
8	(0.9300,0.4658,0.3843)
9	(0.9200,0.0401,0.0020



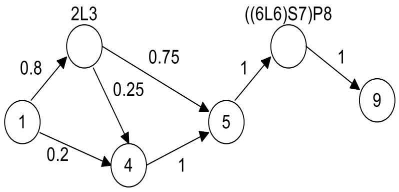
First step: removing the loop patterns

#web service	Q=(readiness,response-time,cost)
1	(0.9600,0.0187,0.0101)
2L3	(0.6185,0.3131,0.0033)
4	(1,0.0392,0.0062)
5	(0.8800,0.1705,0.1151)
6L6	(0.8058,0.0328,8.5244e-04)
7	(0.9600,0.0397,0.0121)
8	(0.9300,0.4658,0.3843)
9	(0.9200,0.0401,0.0020



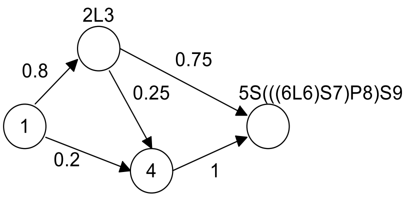
Second step: removing the sequence patterns

#web service	Q=(readiness,response-time,cost)
1	(0.9600,0.0187,0.0101)
2L3	(0.6185,0.3131,0.0033)
4	(1,0.0392,0.0062)
5	(0.8800,0.1705,0.1151)
(6L6)S7	(0.7736,0.0725,0.0129)
8	(0.9300,0.4658,0.3843)
9	(0.9200,0.0401,0.0020)



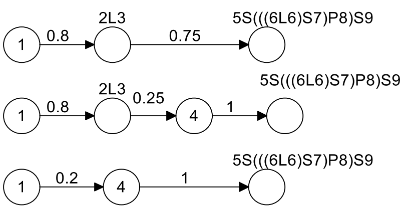
Third step: removing the parallel patterns

#web service	Q=(readiness,response-time,cost)
1	(0.9600,0.0187,0.0101)
2L3	(0.6185,0.3131,0.0033)
4	(1,0.0392,0.0062)
5	(0.8800,0.1705,0.1151)
((6L6)S7)P8	(0.7194,0.4658,0.3972)
9	(0.9200,0.0401,0.0020)



Fourth step: removing the sequence patterns

#web service	Q=(readiness,response-time,cost)
1	(0.9600,0.0187,0.0101)
2L3	(0.6185,0.3131,0.0033)
4	(1,0.0392,0.0062)
5S((6L6)S7)P8)S9	(0.5825,0.6765,0.5143)



Fifth step: extracting the path and then removing the sequence pattern and generating Summary service for each path

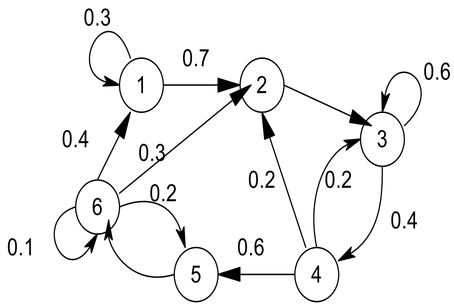
Summary service$_1$= 1S(2L3)S(5S((6L6)S7)P8)S9)	Prob$_1$=0.6	Q' =(0.3458,1.0083,0.5277)
Summary service$_2$= 1S(2L3)S4S(5S((6L6)S7)P8)S9)	Prob$_2$=0.2	Q' =(0.3458,1.0475,0.5339)
Summary service$_3$= 1S4S(5S((6L6)S7)P8)S9)	Prob$_3$=0.2	Q' =(0.5592,0.7344,0.5306)

Example A-3. An example of summarizing the loop pattern with 6 services

In order to make it easier to understand, the beginning of the name of each service is written in find loops. For example, instead of 1L1, only 1 is written in below tables.

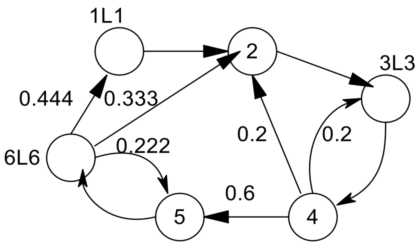
Composition graph

Finding loops from the longest to the smallest ones using Algorithm A-4



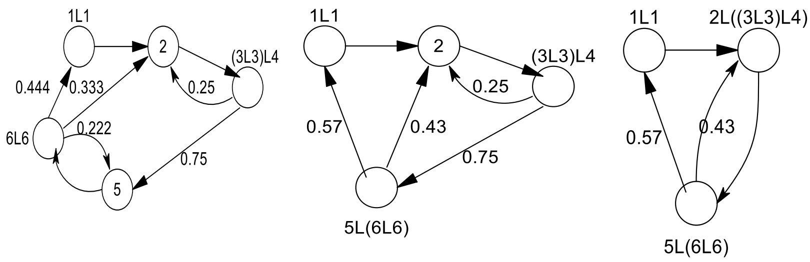
{1,2,3,4,5,6}
{2,3,4,5,6}
{2,3,4}
{5,6}
{3,4}
{6}
{1}
{3}

Composition graph



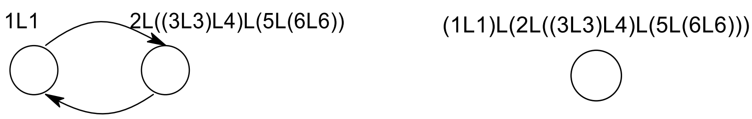
{1,2,3,4,5,6}
{2,3,4,5,6}
{2,3,4}
{5,6}
{3,4}

Composition graph after removing loops of length 1



{1,2,3,5,6}
{2,3,5,6}
{2,3}
{5,6}

{1,2,5}
{2,5}



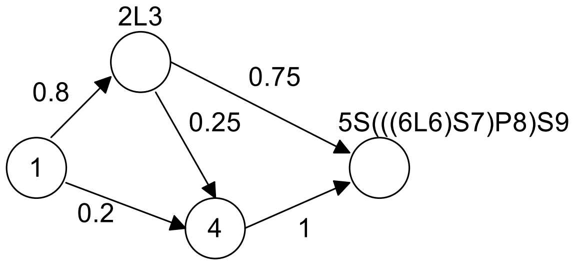
{1,2}	{1}

Composition graph after removing loops of length 2

Example A-4: An example of extracting paths of a composite graph with 6 services after running Algorithm A-5.

This example shows the Array and Path arrays values in each step of the running Algorithm A-5 for extracting the path of the below composition graph.

In order to make it easier to understand, the beginning of the name of each service is written in Array or Path values. For example, instead of 2L3, only 2 is written in below arrays.



Array values								
(1)	(1)	(1,2)	(1,2)	(1,4)	(1,4)	(1,2,4)	(1,2,4)	(1,2,5)
	(1,2)	(1,4)	(1,4)	(1,2,4)	(1,2,4)	(1,2,5)	(1,2,5)	(1,4,5)
	(1,4)		(1,2,4)	(1,2,5)	(1,2,5)	(1,4,5)	(1,4,5)	(1,2,4,5)
			(1,2,5)		(1,4,5)		(1,2,4,5)	
Line2	Lines 5&6	Line 11	Lines 5&6	Line 11	Lines 5&6	Line 11	Lines 5&6	Line 11

Path values	Array values	Path values	Array values	Path values	Array values
(1,2,5)	(1,4,5)	(1,2,5)	(1,2,4,5)	(1,2,5)	
	(1,2,4,5)	(1,4,5)		(1,4,5)	
				(1,2,4,5)	
Line 8	Line 11	Line 8	Line 11	Line 8	Line 11

Example A-5. This example shows the QOSs values used in Algorithm 5 (Prioritization-Solutions function) step by step.

Suppose that the random population selected at the beginning of the algorithm consists of four (Ncom'=4) solutions: Comm'={$(SC,Q')_1$, $(SC,Q')_2$, $(SC,Q')_3$, $(SC,Q')_4$}

where each solution in Comm' includes a pair of SC (a composition of candidate web service) and Q' (QOSs of a Summary service). In the *Prioritization-Solutions function*, we only deal with Q'. We know that each Q'_k contains three quality features (q'_{i1}, q'_{i2} , q'_{i3}). If the QOSs measurements for Comm' solutions are as follows:

Q'_1=(0.79,0.23,0.01)

Q'_2=(0.82,0.42,0.03)

Q'_3=(0.52,0.1,0.1)

Q'_4=(0.67,0.3,0.008)

After running line 4 of Algorithm 4 and calling *Prioritization-Solutions* function, the sorted1 set is sorted descending on q'_{i1}, and the sorted2 and sorted3 sets are sorted ascending on q'_{i2} and q'_{i3}.

sorted1={q'_{21}, q'_{11}, q'_{41}, q'_{31}}

sorted2={q'_{32}, q'_{12}, q'_{42}, q'_{22}}

sorted3={q'_{43}, q'_{13}, q'_{23}, q'_{33}}

$sortedComm_1$= (2+2+2)/3= 2, /* for Q'_1,

$sortedComm_2$= (1+4+3)/3= 2.6, /* for Q'_2

$sortedComm_3$= (4+1+4)/3 = 3, /* for Q'3

$sortedComm_4$= (3+3+1)/3 = 2.3 /* for Q'_4

NewComm'={$(SC,Q')_1$, $(SC,Q')_4$, $(SC,Q')_2$, $(SC,Q')_3$}

Example A-6: This example shows the parameters values used in Algorithm 6 (*Discretization-Solutions function*) step by step.

Suppose that $3\in CWS_1$, $600\in CWS_2$, $1400\in CWS_3$, $2300\in CWS_4$ (see Def .2), consider these candidates web service for application graph with 4 function f_1, f_2, f_3, f_4 respectively and itomega=1. So, we have:

$SC_1=(w_\omega)_1=$ {3, 600, 1400, 2300} where these candidates 3, 600, 1400, 2300 are selected from CWS_1, CWS_2, CWS_3, CWS_4, respectively for running f_1, f_2, f_3, f_4 functions. SC_1.Q= $(w_\omega)_1$.Q ={(0.82,0.2,0.032), (0.75,0.1,0.001), (0.68,0.15,0.003), (0.78,0.3,0.002)} are the QOSs for these candidate services.

In the same way, for alpha, beta and delta wolves (solutions) we also have:
w_α ={50, 400, 1100, 2100} and w_α.Q ={(0.92,0.32,0.005), (0.73,0.4,0.006), (0.88,0.2,0.002), (0.96,0.5,0.005)}

w_β ={20, 740, 1800, 2050} and w_β.Q ={(0.88,0.2,0.002), (0.64,0.35,0.04), (0.93,0.1,0.002), (0.62,0.1,0.005)}

w_δ={210, 531, 1920, 2150} and w_δ.Q ={(0.96,0.5,0.07), (0.62,0.01,0.006), (0.89,0.18,0.02), (0.82,0.25,0.009)}

The QOS of the new omega wolf (generated in line 10 algorithm 4) in continuous space are as follows.
sNewOmegaContinues ={(0.87,0.23,0.006),(0.59,0.018,0.008),(0.91,0.3,0.0012),(0.71,0.1,0.008)}

Now we want to find a concrete candidate for sNewOmega wolf in the discrete space. So, we have: sNewOmega={?, ?, ?, ?}

Based on the proposed *Discretization-Solutions* function, if the QOS of i^{th} dimension of sNewOmegaContinues are within the specified range (less than 1), it is compared with QOS of i^{th} dimension of delta, beta, alpha, and if dominated by any of these wolves, the dominant wolf candidate is selected. In this example, the QOS of the first dimension of the new wolf in the continuous space (sNewOmegaContinues) is (0.87, 0.23, 0.006), which is first compared with the QOS of the first dimension of the delta wolf (0.96,0.5,0.07), the delta wolf can't dominate it, so it is compared to beta wolf (0.88,0.2,0.002). The QOS of beta wolf are (0.88, 0.2, 0.002), so the beta wolf (with candidate 20) dominates the sNewOmegaContinues wolf in the first dimension. So, the first candidate of beta wolf is selected for the first dimension of the sNewOmega wolf in the discrete space

sNewOmega={20, ?, ?, ?} and sNewOmega.Q={(0.88,0.2,0.002), (?), (?), (?)}

The QOS of the second dimension of the sNewOmegaContinues wolf (0.59, 0.018, 0.008) are compared with delta wolf (0.62, 0.01, 0.006) and delta wolf dominates it. Therefore, the candidate for the second dimension of the sNewOmega wolf is also determined.

sNewOmega={20, 531, ?, ?} and sNewOmega.Q={(0.88,0.2,0.002), (0.62,0.01,0.006), (?), (?)}

The QOS of the third dimension of the sNewOmegaContinues wolf (0.91, 0.3, 0.0012) with the QOSs of delta (0.89, 0.18, 0.02), beta (0.64, 0.35, 0.04) and alpha (0.88, 0.2, 0.002) is compared and none of them dominates the sNewOmegaContinues wolf. So, all available candidates for this web service are reviewed. The first candidate that could dominate the sNewOmegaContinues wolf will be selected. For example, the QOS values of candidate 1990 dominated the QOS values of the sNewOmegaContinues wolf. So,

sNewOmega={20, 531, 1990, ?}and sNewOmega.Q={(0.88,0.2,0.002), (0.62,0.01,0.006), (0.93,0.3,0.0012), (?)}

Now the QOS of the last dimension of sNewOmegaContinues wolf (0.71,0.1,0.008) with the QOS of delta (0.82,0.25,0.009), beta (0.62,0.1,0.005) and alpha (0.96,0.5,0.005) is compared and none of them dominates this dimension of the sNewOmegaContinues wolf. So, all available candidate indexes for this web service are reviewed and none of them dominate it. So, the last dimension candidate of current wolf, (0.78,0.3,0.002) is determined for the last dimension of sNewOmega wolf in the discrete space. So, we have:

sNewOmega={20, 531, 1990, 2300} and sNewOmega.Q={(0.88,0.2,0.002), (0.62,0.01,0.006), (0.93,0.3,0.0012), (0.78,0.3,0.002)}

if the QOS of i^{th} dimension of sNewOmegaContinues are out of the specified range (more than 1), select a candidate randomly and placed into the i^{th} dimension of sNewOmega wolf.

Appendix 3. Fig. A-1.

The flowchart of ***proposed MOEA***