

# A GP Approach to QoS-Aware Web Service Composition including Conditional Constraints

Alexandre Sawczuk da Silva, Hui Ma, Mengjie Zhang

**Evolutionary Computation Research Group**

School of Engineering and Computer Science, Victoria University of Wellington

*IEEE Congress on Evolutionary Computation, 25-28 May 2015*

# Introduction

**Service-Oriented Architecture (SOA):** Organise processes and data in reusable modules for integration into new applications.

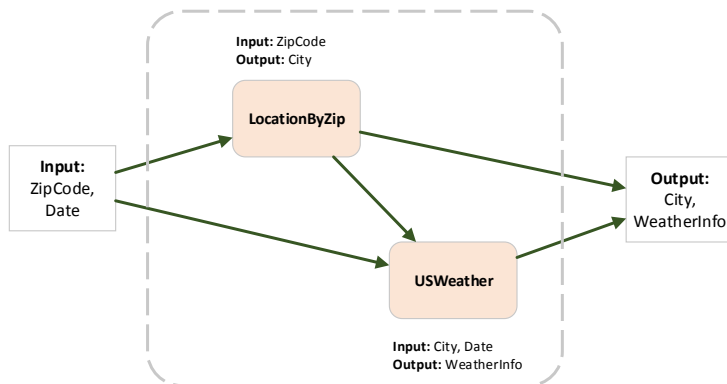


## Web service

A functionality module that provides operations accessible over the network via a standard communication protocol.

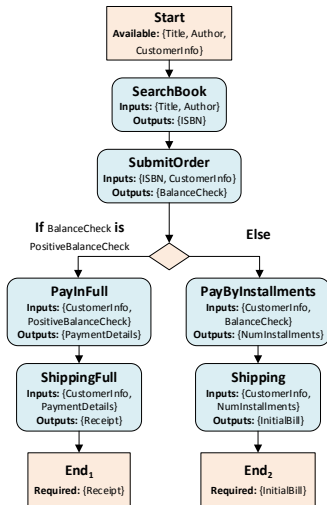
# Web Service Composition

The combination of Web services to achieve a more complex task.  
Fully automated scenario:



New weather by zip code service

# A Composition Example with Branching



Certain compositions require alternative paths according to runtime values.

**Example:** Depending on balance, pay in full or pay in installments.

# Composition Dimensions

- 1 Functional correctness:** Service inputs and outputs must be properly linked (e.g. *FourDigitNumber* → *ZipCode*, but not *FourDigitNumber* → *City*).
- 2 Conditional constraints:** Condition leading to multiple possible execution paths (e.g. if *City* is a *NewZealandCity*, produce *WindForecast* instead of *GeneralForecast*).
- 3 Quality of Service (QoS):** The overall quality of the composition (e.g. *lowest execution time*, *lowest cost*).

# Existing Approaches

## AI Planning

Build a solution service by service.

Dimensions: *Functional correctness, conditional constraints.*

## Evolutionary Computation (EC)

Improve population of solutions over multiple generations.

Dimensions: *Functional correctness, QoS.*

## Hybrid Approaches

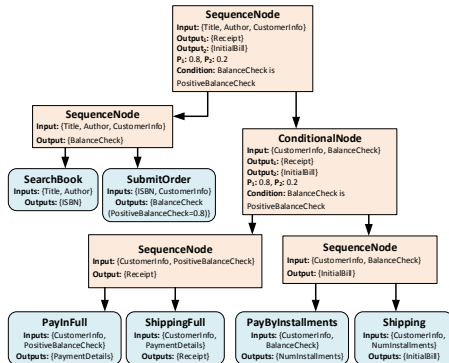
Combine AI planning and EC ideas.

Dimensions: *Functional correctness, QoS.*

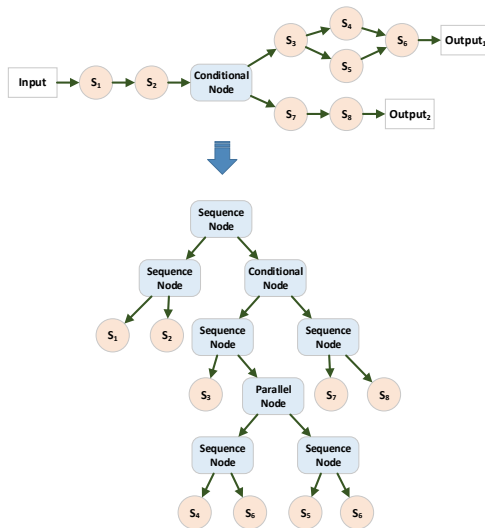
# Goal

To propose a Genetic Programming (GP) composition approach that simultaneously considers all dimensions.

- 1 Trees preserve functional correctness.
- 2 Conditions encoded in trees.
- 3 Optimisation performed on QoS.



# Candidate Representation



- Tree equivalent to graph composition.
- Parallel, sequential, and conditional represented as non-terminal nodes.
- Candidate services as terminal nodes.



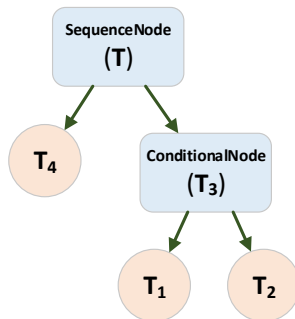
# Population Initialisation

An algorithm is used to create a candidate in graph format, and then translate it into a tree representation.

```

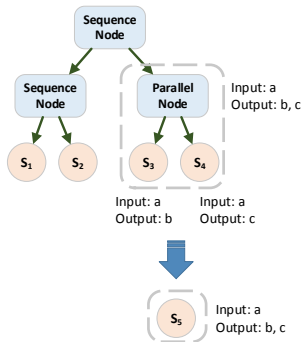
Input :  $I, O_1, O_2, C, P$ 
Output: candidate tree  $T$ 
1: if  $O_2 \neq \emptyset$  then
2:    $G_1 \leftarrow \text{createGraph}(I \cup C.\text{if}, O_1)$ ;
3:    $G_2 \leftarrow \text{createGraph}(I \cup C.\text{else}, O_2)$ ;
4:    $T_1 \leftarrow \text{toTree}(G_1.\text{input})$ ;
5:    $T_2 \leftarrow \text{toTree}(G_2.\text{input})$ ;
6:    $T_3 \leftarrow \text{new ConditionalNode}(C)$ ;
7:    $T_3.\text{leftChild} \leftarrow T_1$ ;
8:    $T_3.\text{rightChild} \leftarrow T_2$ ;
9:   if  $C \sqsubseteq I$  then
10:     $T_3.\text{prob} \leftarrow P$ ;
11:    return  $T_3$ ;
12:   else
13:     $G_4 \leftarrow \text{createGraph}(I, C.\text{else})$ ;
14:     $T_4 \leftarrow \text{toTree}(G_4.\text{input})$ ;
15:     $T_3.\text{prob} \leftarrow T_4.\text{final}.P$ ;
16:     $T \leftarrow \text{new SequenceNode}()$ ;
17:     $T.\text{leftChild} \leftarrow T_4$ ;
18:     $T.\text{rightChild} \leftarrow T_3$ ;
19:    return  $T$ ;
20:   end
21: else
22:    $G \leftarrow \text{createGraph}(I, O_1)$ ;
23:    $T \leftarrow \text{toTree}(G.\text{input})$ ;
24:   return  $T$ ;
25: end

```

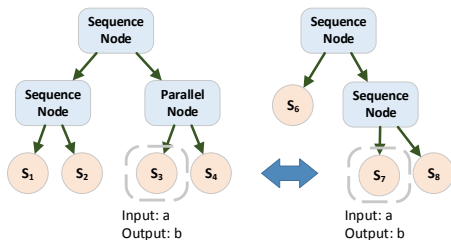


# Mutation and Crossover

**Mutation:** Selects random node and replaces it with equivalent subtree.



**Crossover:** Swaps any two equivalent terminal nodes.

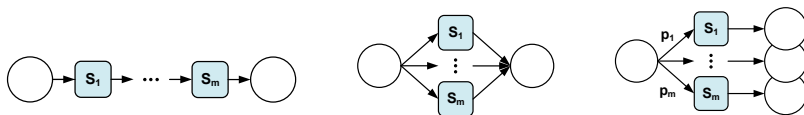


# Fitness Function

Measures the overall quality of a composition candidate (minimising).

$$fitness_i = w_1(1 - A_i) + w_2(1 - R_i) + w_3 T_i + w_4 C_i$$

$$\text{where } \sum_{i=1}^4 w_i = 1$$



# Experiments

- Lack of datasets supporting composition with branching.
- Lack of comparable approaches that produce solutions with multiple output possibilities.

**Decision:** Create datasets, execute for conditional compositions and also for each branch separately.

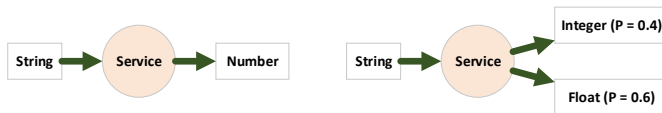
Parameters:

Independent runs	50	Elitism candidates	1
Population size	20	Tournament size	7
Crossover probability	0.9	Fitness weights	0.25 (all)
Mutation probability	0.1		

# Creation of Datasets

Modified from WSC2008.

- Can be extended to contain QoS.
- Provides ontology of input and output values.



Tasks requiring branching were created.

# Results

Set (size)	Conditional	
	Avg. fitness	Avg. time (s)
1 (158)	0.601 $\pm$ 0.013	1.290 $\pm$ 0.100
2 (558)	0.712 $\pm$ 0.009	2.829 $\pm$ 0.250
3 (604)	0.631 $\pm$ 0.008	13.285 $\pm$ 1.229
4 (1041)	0.718 $\pm$ 0.048	6.146 $\pm$ 0.574
5 (1090)	0.698 $\pm$ 0.005	11.759 $\pm$ 0.948
6 (2198)	0.662 $\pm$ 0.017	92.392 $\pm$ 11.353
7 (4113)	0.578 $\pm$ 0.010	97.344 $\pm$ 13.705
8 (8119)	0.656 $\pm$ 0.005	326.387 $\pm$ 37.659

Set (size)	Non-conditional			
	If branch		Else branch	
	Avg. fitness	Avg. time (s)	Avg. fitness	Avg. time (s)
1 (158)	0.508 $\pm$ 0.000	0.563 $\pm$ 0.144	0.588 $\pm$ 0.037	0.718 $\pm$ 0.079
2 (558)	0.588 $\pm$ 0.084	1.490 $\pm$ 0.526	0.694 $\pm$ 0.016	1.527 $\pm$ 0.194
3 (604)	0.365 $\pm$ 0.000	4.387 $\pm$ 0.768	0.788 $\pm$ 0.000	7.099 $\pm$ 0.906
4 (1041)	0.689 $\pm$ 0.064	4.510 $\pm$ 1.177	0.741 $\pm$ 0.429	3.568 $\pm$ 0.429
5 (1090)	0.446 $\pm$ 0.000	5.726 $\pm$ 0.755	0.688 $\pm$ 0.011	6.491 $\pm$ 0.743
6 (2198)	0.412 $\pm$ 0.063	58.295 $\pm$ 12.765	0.645 $\pm$ 0.024	52.308 $\pm$ 5.785
7 (4113)	0.363 $\pm$ 0.000	44.838 $\pm$ 5.926	0.688 $\pm$ 0.032	51.725 $\pm$ 4.260
8 (8119)	0.474 $\pm$ 0.000	106.119 $\pm$ 7.152	0.766 $\pm$ 0.002	186.896 $\pm$ 20.008

# Conclusions

Novel approach addresses **three composition dimensions** simultaneously (fully functional, contain branches, quality-optimised).

- Solutions found with similar performance as non-branching technique.

**Future work:** More than two branches, more complex branching conditions, analysis of convergence behaviour.

Thank you!

Questions?