BENNS: A SURROGATE FOR HYBRID ONLINE-OFFLINE EVOLUTION OF OPTIMAL SERVICE FUNCTION CHAIN EMBEDDING

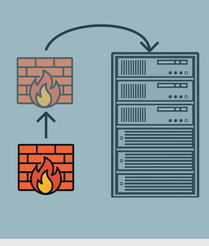
Theviyanthan K. hello@thivi.dev
School of Computing Science, University of Glasgow

Service Function Chaining (SFC)

SFC combines NFV and SDN to create a service overlay over the physical network to make computer networks programmable. This takes us a step closer to autonomous networks.

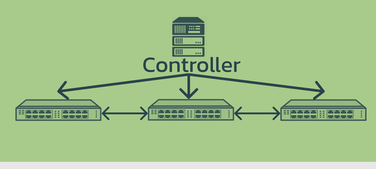
Network Function Virtualisation (NFV)

NFV virtualises network functions such as firewalls and embeds them on servers (hosts).



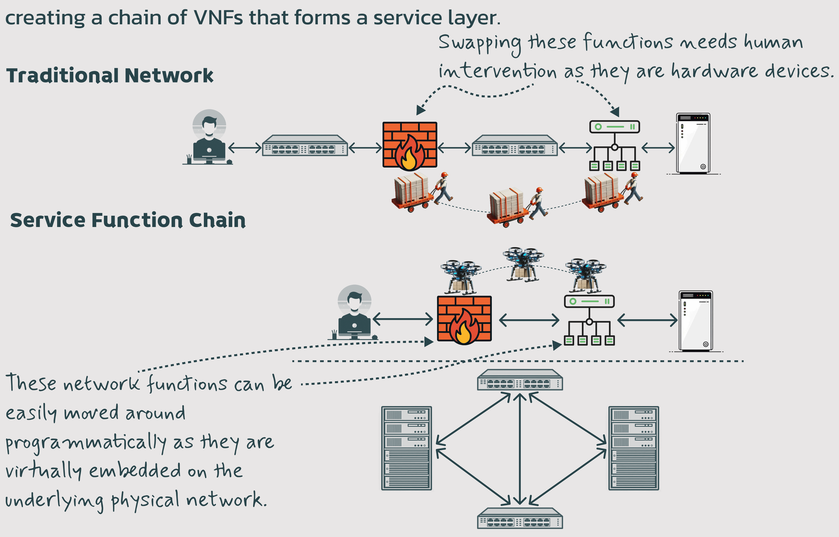
Software Defined Networking (SDN)

SDN separates the control plane of network switches from the data plane allowing network paths to be programmed by a controller.



Service Layer

The Virtualised Network Functions (VNFs) produced by NFV are linked together using SDN, creating a chain of VNFs that forms a service layer.

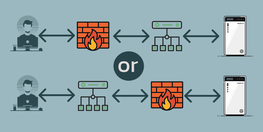


SFC Embedding Challenges

Optimally embedding SFCs on the physical network is NP-hard. It consists of three sub-problems. Studies show that to produce an optimal solution, all three sub-problems have to be optimised simultaneously, rather than sequentially.

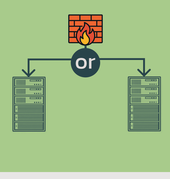
Chain Composition

How to optimally order the VNFs in an SFC?



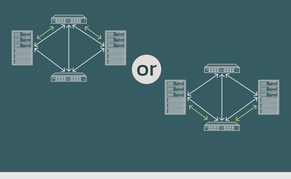
VNF Embedding

How to optimally embed VNFs on hosts?



Link Embedding

How to optimally route traffic between VNFs?



Optimisation Goals

Maximise Acceptance Ratio

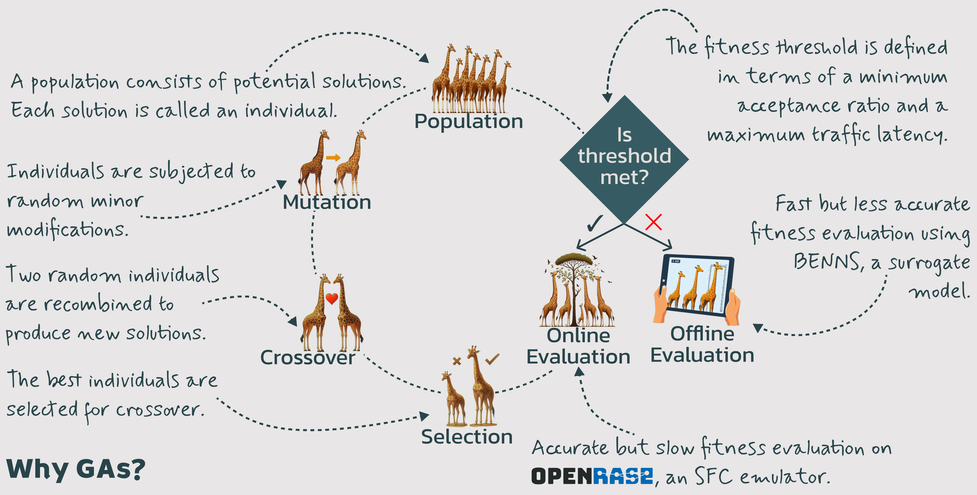
Acceptance Ratio is the number of SFCs that can be embedded over the number of requests received to embed SFCs.

Minimise Traffic Latency

Traffic latency is the round-trip time traffic takes to traverse SFCs. This is experimentally measured or approximated.

Genetic Algorithms (GAs)

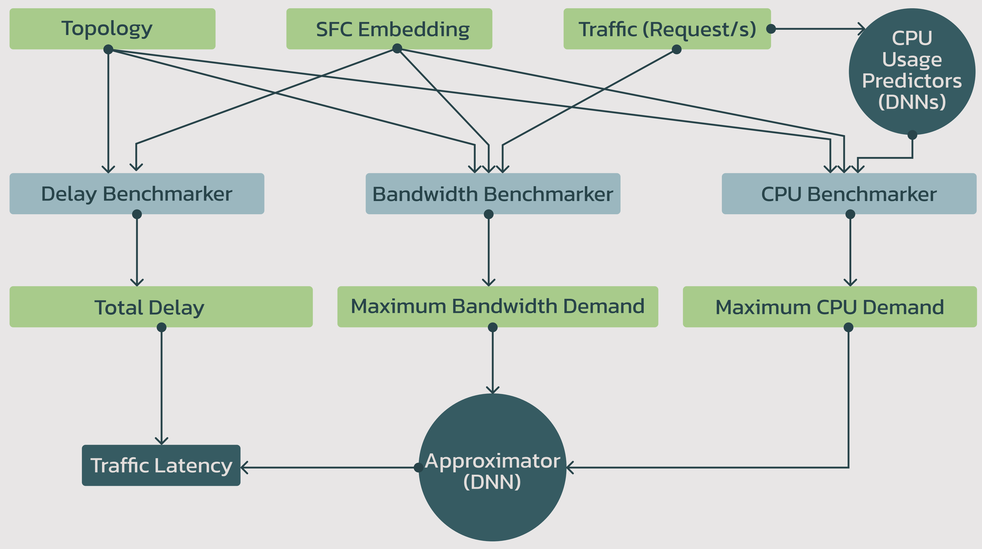
GAs are metaheuristic algorithms that can optimise NP-hard problems. They apply the idea of evolution to optimise complex problems by evolving potential solutions iteratively.



Why GAs?

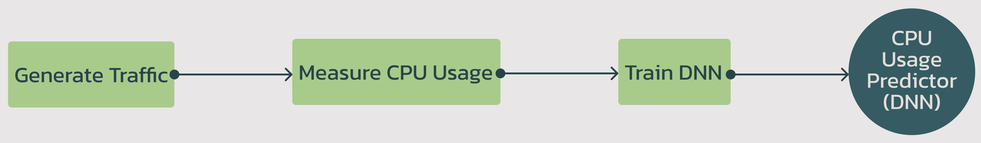
Only 19 of the 209 studies used GAs.	GAs adapt to dynamic network environments.	GAs don't need a fitness function gradient.	GAs can search concurrently.

BENNS Surrogate Model



CPU Usage Predictors

These are Deep Neural Networks (DNNs) that predict the CPU usage of a VNF for a given number of requests per second. We have a DNN for each VNF available in the environment. The training data is generated by profiling the VNFs with synthetic traffic.



Benchmarkers

We use benchmarkers to encode SFCs and network topologies, which are complex, into numerical values that can be used to train a DNN model to predict the traffic latency.

Delay Benchmarker

Sums the transmission delay of each physical link that links the VNFs of an SFC and outputs the total transmission delay.

Bandwidth Benchmarker

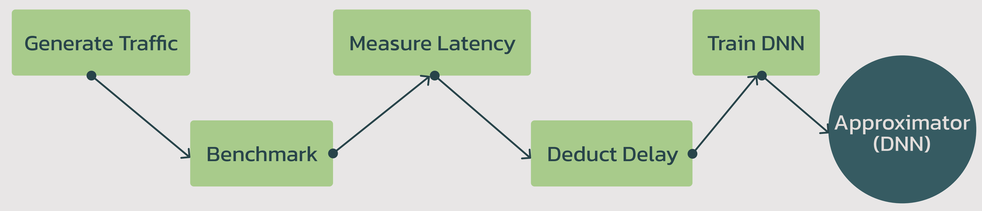
Finds the bandwidth demand of each physical link used by an SFC by dividing the total requests traversing the link per second by its bandwidth and outputs the maximum bandwidth demand.

CPU Benchmarker

Finds the CPU demand of each host used to host the VNFs of an SFC by dividing the total predicted CPU usage by the no. of CPUs available and outputs the maximum CPU demand.

Approximator

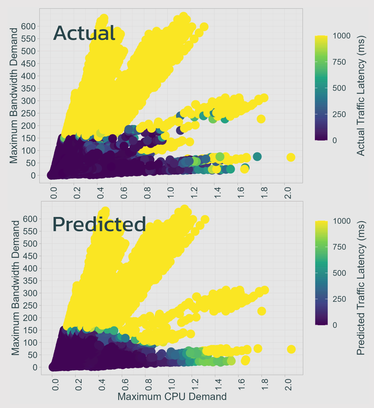
The approximator is a DNN consisting of 2 hidden layers with 16 neurons each. The neurons use the Sigmoid activation function. We ran 11 different experiments to generate data to train the approximator.



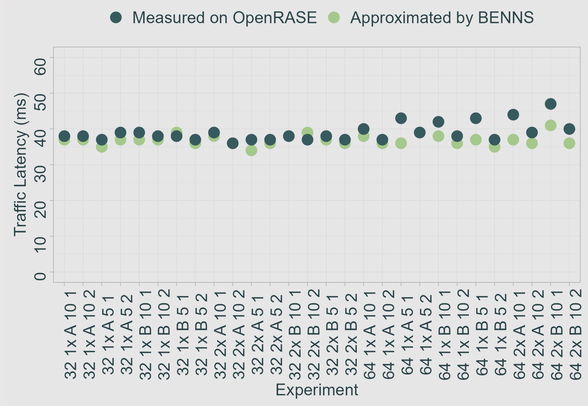
Evaluation & Conclusion

Evaluation of Accuracy of BENNS

On Validation Dataset

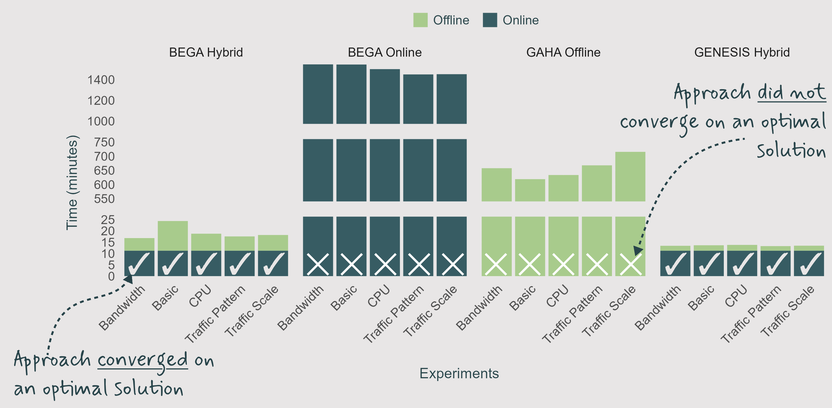


Across 28 Different Experiments



Evaluation of Hybrid Evolution

We evaluated our hybrid online-offline approach using two GAs, namely BEGA and GENESIS, and compared the hybrid approach to an online-only approach using BEGA and an offline-only GA approach called GAHA across 5 experiments with different network configurations.



Conclusion

Hybrid online-offline evolution using BENNS can optimally embed SFCs across different network scenarios by enabling greater exploration without sacrificing speed.