Impact of Fluctuating Goals on Adaptability of Evolvable VNF Placement Method

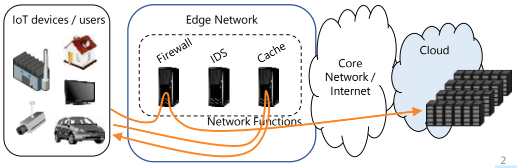
Mari Otokura[1], Kenji Leibnitz[2], Yuki Koizumi[1], Daichi Kominami[1], Tetsuya Shimokawa[2], Masayuki Murata[1]

[1] Osaka University, Japan

[2] NICT, Japan

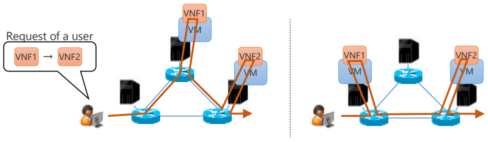
Research Background

- Communication services are becoming more diverse and dynamic
  - Internet of Things (IoT) is currently being realized
- Communication service providers will offer flexible and dynamic *network functions*
  - Network functions process packets in the "middle" of networks
    - Examples: firewalls, intrusion detection systems (IDS), caches
  - Network functions have been implemented in hardware
    - Not flexible and not dynamic



Network Function Virtualization (NFV)

- NFV implements network functions in software
- Virtual Network Functions (VNFs):
  - Network functions virtualized by NFV
  - Run on virtual machines (VMs)
- VNF placement problem
  - Decision of VNF placement on physical machines (PMs) in physical networks



Large delay

Small delay

System Model

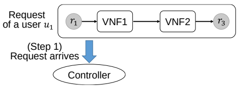
(Step 1) *Request* from user for a *VNF chain* arrives

(Step 2) Controller splits VNFs of chain into *components*

- Components: small VNF software modules [4]

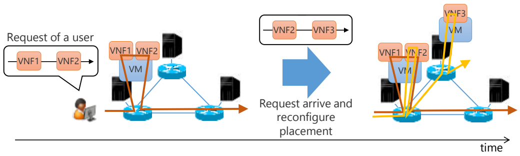
(Step 3) Controller places components on PMs and assigns cores to them

(Step 4) Controller decides routes of traffic for chains through required components



Dynamic VNF Placement Problem

- Reconfiguration of VM/VNF placements on physical network when requests for VNF chains arrive/depart
- Main requirement for placement computation: short calculation time
- Solving optimization problem at every request change:
  - Difficult to realize since even the static VNF placement problem is NP-hard



Objective

- We previously proposed an evolutionary method for dynamic VNF placement problems named Evolvable VNF Placement (EvoVNFP) [12]
  - Utilizing knowledge from biological evolution under varying environments
    - When organisms evolve in varying environments:
      - Organisms become robust to environmental changes [13]
      - Evolution of organisms speeds up [14]
- Objective
  - Evaluating EvoVNFP in greater detail to clarify the influence of the parameter settings on the performance

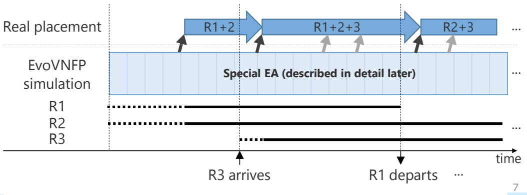
[12] M. Otokura, K. Leibnitz, Y. Koizumi, D. Kominami, T. Shimokawa, and M. Murata, "Application of Evolutionary Mechanism to Dynamic Virtual Network Function Placement," in Proceedings of ICNP Workshop on Control Operation and Application in SDN protocols (CoolSDN), Nov. 2016.

[13] N. Kashtan and U. Alon, "Spontaneous Evolution of Modularity and Network Motifs," *PNAS*, vol. 102, no. 39, pp. 13773–13778, Sep. 2005.

[14] N. Kashtan, E. Noor, and U. Alon, "Varying Environments Can Speed Up Evolution," *PNAS*, vol. 104, no. 34, pp. 13711–13716, Aug. 2007.

Evolvable VNF Placement (EvoVNFP)

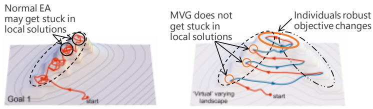
- Dynamic VNF placement method addressing dynamic arrivals/departures of VNF chain requests
  - Calculation of placements by a special type of Evolutionary Algorithm (described in detail later)
  - When simulations generate placements which meet predetermined objectives, the controller implements these generated placements as real ones



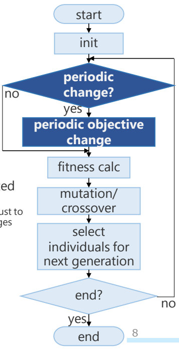
[13] N. Kashtan and U. Alon, "Spontaneous Evolution of Modularity and Network Motifs," *PNAS*, vol. 102, no. 39, pp. 13773–13778, Sep. 2005.

Modularly Varying Goals (MVG) [13]

- An optimization algorithm as extension of EA, which imitates biological evolution in varying environments
- MVG changes its objective regularly every fixed number of generations
- Effects of regular changes:
  - Individuals become robust to objective changes
  - Evolution itself speeds up
    - Because getting stuck in local solutions is prevented


Normal EA may get stuck in local solutions
Goal 1
start
MVG does not get stuck in local solutions
Individuals robust to objective changes
'Virtual' varying landscape
start

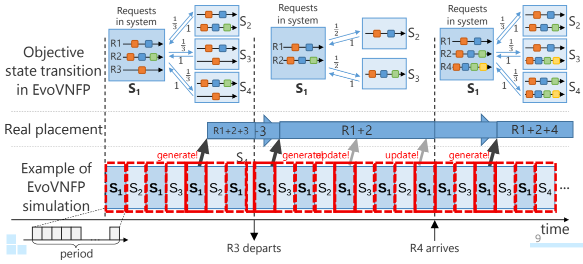
Conventional EA MVG



[14] N. Kashtan, E. Noor, and U. Alon, "Varying Environments Can Speed Up Evolution," *PNAS*, vol. 104, no. 34, pp. 13711–13716, Aug. 2007.

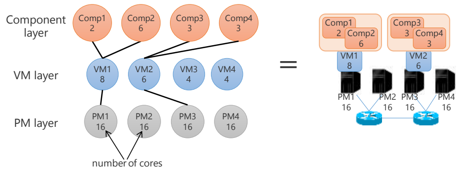
Detailed Behavior of EvoVNFP

- Intentionally change objectives every fixed number of generations (= period)
- Intentionally use EA without re-initialization of population when objectives change



Individuals and Mutations

- Example structure of an individual (see figure below):
  - Individual represents placement in network
    - Connection between VM layer and component layer: allocation of component on VM
    - Connection between PM layer and VM layer: allocation of VM on PM
- Mutation: randomly change one element of an individual
  - Change connections or the number of cores saved in nodes



Fitness Function

- Evaluate how well placements adapt to objectives
  - If individuals can be converted to placements:
    - Small average delay of chains and small number of used cores → high fitness
  - Otherwise:
    - Fitness is a negative value
    - Small number of elements in individuals violating the constraints → high fitness

$$\boxed{F = \begin{cases} \left(\dfrac{\hat{d}}{d_{max}} + \dfrac{W(\sum_{i,k} m_{i,k})}{c_{max}}\right)^{-1} & \text{if individuals can be converted to placements} \\ -Z & \text{otherwise} \end{cases}}$$

d_{max}: reference value of delay
c_{max}: maximum number of cores
Z: number of elements which violate the constraints

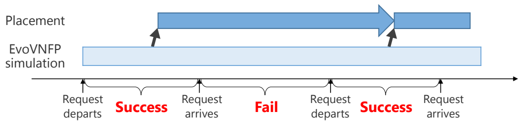
Simulation Settings

- Physical network: 5 routers, 10 PMs, each PM has 16 cores
- Requests: tuples consisting of ingress router, egress router, VNF chain, and transmission rate
  - Example: $(r_1,\ r_3,\ \{VNF1 \rightarrow VNF2\},\ 200\ \text{Mbps})$
- Reference methods for comparison:
  - Conventional EA (Conv): normal EA that is rerun whenever there is an arival/departure of requests
  - Random Immigrant GA (RandImm) [15]
    - RandImm initializes randomly selected individuals after mutation step
- Parameters:
  - Population size: 1000, elite size: 100, mutation probability: 0.8
  - Replacement rate (RandImm): 0.3

[15] J. Grefenstette, "Genetic Algorithms for Changing Environments," in *Proceedings of PPSN 1992*. Elsevier, Sep. 1992, pp. 137–144.

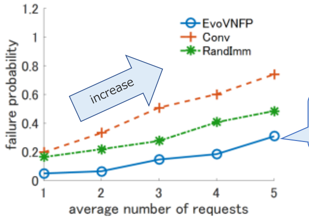
Types of Evaluations and Metrics

- Evaluations under varying parameters
  - Evaluation with varying system load (EvoVNFP, Conv, RandImm)
  - Evaluation of different period lengths (EvoVNFP)
- Evaluation metric
  - Failure probability
    - Probability of finding no feasible solution until the next objective change



Evaluation with Varying System Load

- Failure probability of EvoVNFP is lowest
- Failure probability increases when the average load increases

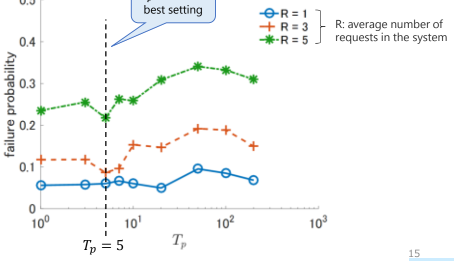


EvoVNFP has the lowest failure probability

Evaluation of Different Period Lengths

- $T_p = 5$ is the best setting for the considered simulation setting

$T_p = 5$ is the best setting



R: average number of requests in the system

Summary and Future Work

- Summary
  - Evaluating dynamic VNF placement method named EvoVNFP in greater detail by computer simulations
    - EvoVNFP generates placements which meet user requests by MVG
      - When requests arrive/depart, EvoVNFP runs EA without reinitializing population
      - EvoVNFP switches between real objectives and relaxed sub-objectives every fixed number of generations
    - Evaluation of EvoVNFP by computer simulations
      - EvoVNFP can follow the dynamics of the request arrival/departure
      - Specific parameter settings of EvoVNFP make its adaptability even better
- Future work
  - Application of further evaluation metrics