A Greedy Heuristic Deployment strategy for VNFs

Background

Expensive equipment/p er

Difficult to add new featu

Difficult manage

Cannot be scaled on demand

Middlebox which is playing a crucial role in current net provide plentiful network functions.

Hardware middleboxes drawbacks.

By separating network function from the underlying hardware, the deployment will be more flexible.
How to deploy dynamic, flexible and robust service function is an attractive and useful problem.

firewall

DDoS protection

IDS

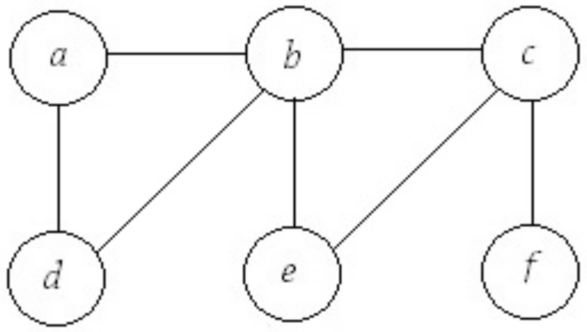
ad insertion

BRAS

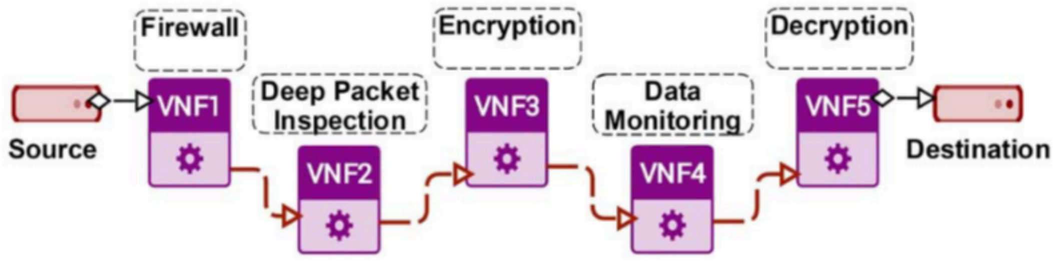
Resource Scheduling for Network Function Virtualization

- Instance: a physical network $G^s = (V^s, E^s)$ with fixed node resources and edge resources. A service function chain set.
- Find an optimal strategy to deploy the SFC set in the physical network .

A physical network instance:



A Service Function Chain instance :$R_i = \left(r_1^i, r_2^i, r_3^i, \dots r_n^i\right)$ from source to destination.



A service function chain

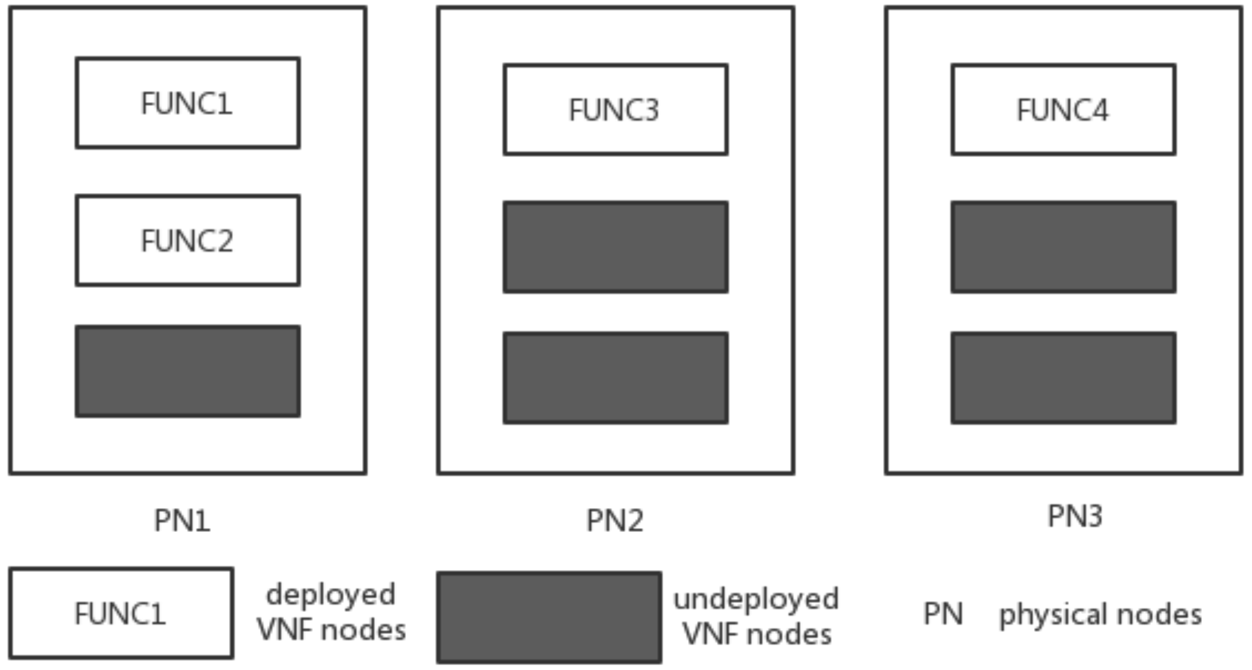
Related work

- Resources Allocation of SFC
- Virtual Machine Deployment
- Virtual Network Embedding

Resources Allocation of SFC

Input: a physical network and a SFC set.
Output: a network function deployment .

Differences:
1. A physical node can be deployed with many functions.
2. No share function node.
3. Fixed requirements.

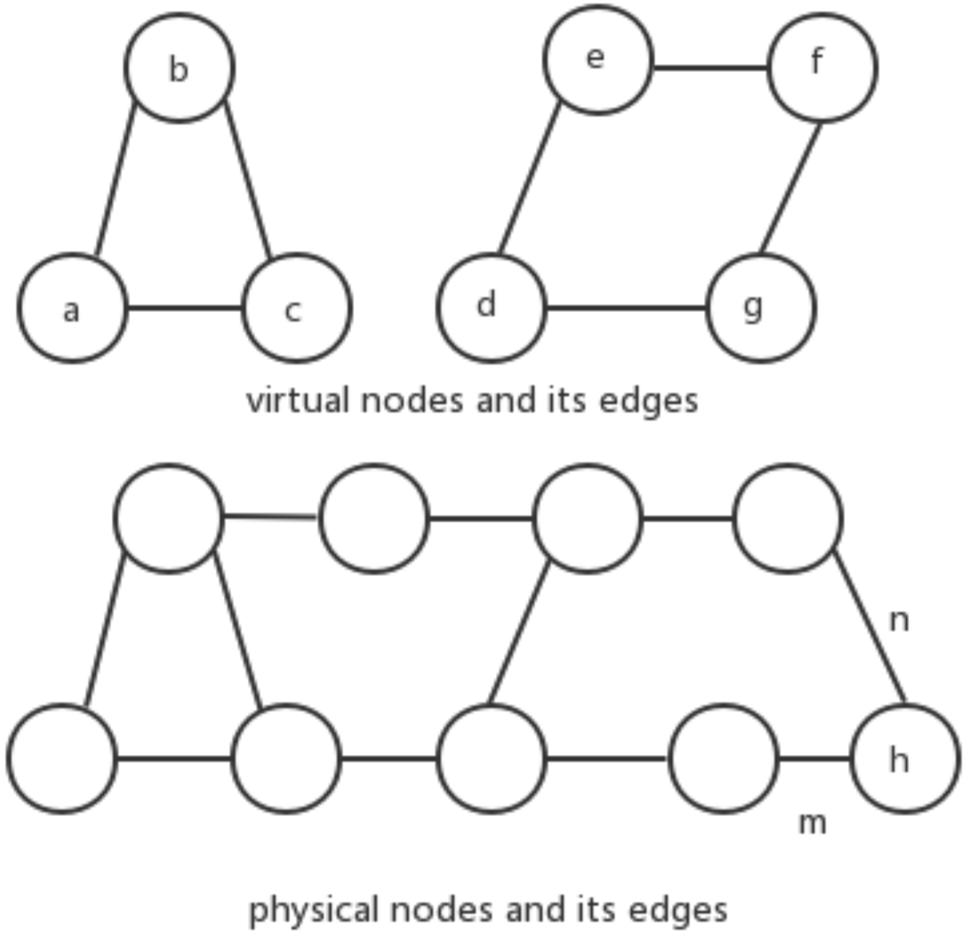


Virtual Machine Deployment

- Requirements are deployed in the virtual machine, and virtual machine is deployed in the physical node.
- The problem of virtual machine deployment is to find a suitable place for virtual machine.
- This problem is the next step of the talking problem.

Virtual Network Embedding

- Embedded a virtual network to a physical network
- It also take allocation of node resources and edge resources into account.
- Differences: it is an undirected cyclic graph.



- For multi-service chains, to find a optimal deployment plan is a NP-hard problem. Consider the service chain is queued to be deployed.
- When the service chain requires, first judge whether the resources is enough.
- Under common circumstances, a virtual machine can be deployed with a VNF function, and different service chains can share the same VNF function.

$$Maximize: \quad N_R^S$$
$$Minimize: \quad Cost(G^S, R, M)$$

Deployment mechanisms

- Deployment of functions

optimal nodes set will be selected to load all the required VNF because virtual functions of different nodes in the physical network are different.

- Selection of path

after selection of nodes, a proper path between nodes is also needed to make one node corresponding with another and achieve a splendid working efficiency.

For an incoming VNF chain requests, we need to map all the VNFs of the service chain and construct a physical path to link all the nodes.

For an incoming VNF chain request, there are 2 choices.

- Select a new node where no VNF is deployed.
- Select a rest-computing-power-enough node where a same VNF has already been deployed.

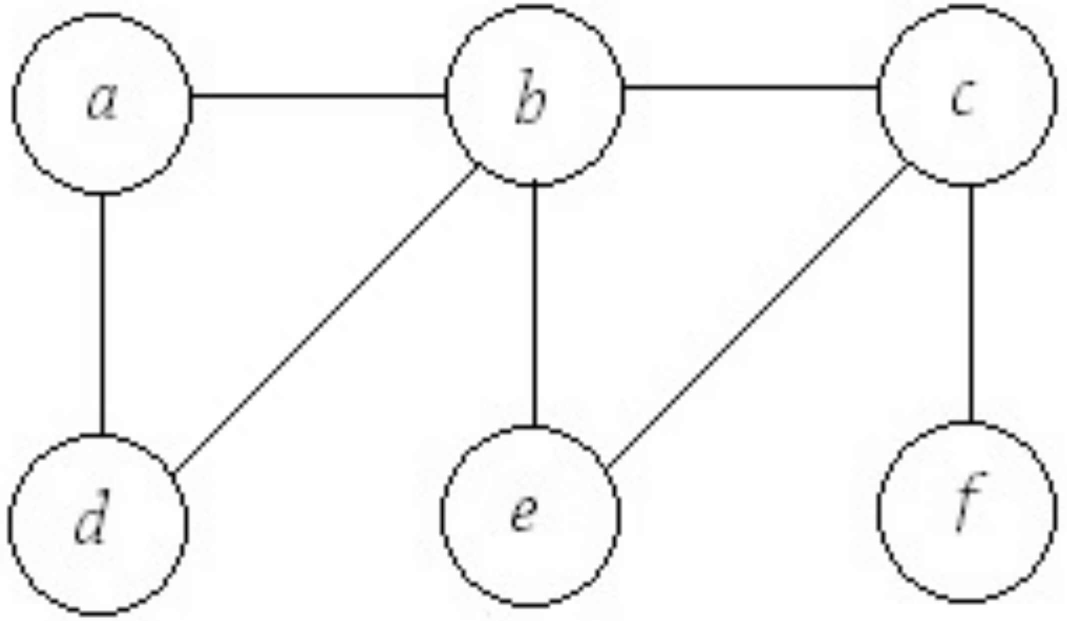
if the first choice is used, the whole physical nodes number is high which will increase the cost of nodes of the whole network.

If the second choice is used, the cost of the edge will increase.

How to balance reused nodes and bandwidth costs is a considerable problem!

Example

For a SFC R_i, which has A,B and C functions.
Node d has been deployed with A, node e has been deployed with B, node f has been deployed with C.



How to deployed SFC R_i ?

Dynamic requirements

- The computing resources which is required by a virtual network function is not definite.

$$tuple(r_k^i) < basic_i^k, variable_i^k, p_i^k >$$

The required resources consist:

1.Basic requirement ;

2.Floating requirements; in a possibility of p.

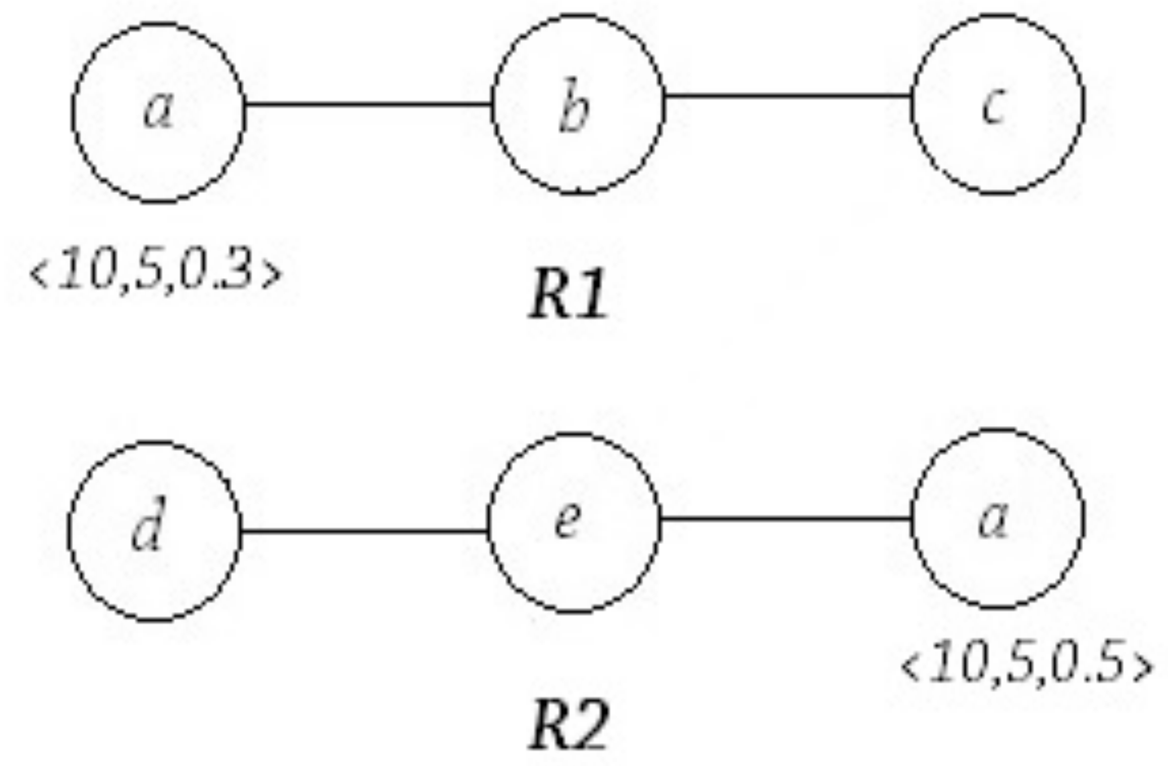
How to judge the share

$$RestCp(v_i) \geq \sum_i^j basic_i^k + variable_i^k$$

$$p_{v_i}^{R_i} * p_{v_i}^{R_j} < p^{th}$$

Example

the fixing requirement of node **a** in R1 is 10 and the floating requirement of node a is 5 and the possibility is 0.3. the fixing requirement of node **a** in R2 is 10 and the floating requirement of node a is 5 and the possibility is 0.3. If they want to deploy in a node with 25 computing resources.



$$0.3 \times 0.5 = 0.15 < 0.2;$$

Can be shared

How to reuse physical nodes?

- X represent the number of node which will be reused for chain $\boldsymbol{R_i}$.
- X is dynamic.
- *l(x) the length of the edge when reusing the x nodes.*
- *l(x) is depending on x.*

Find reused factor x

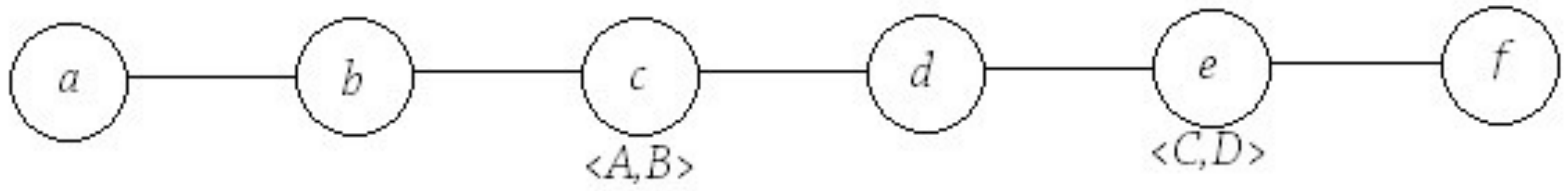
$$\text{maximize: } n$$

$$\textbf{\textit{s.t.:}} \quad x \in \{0,1,2,\dots \bar{x}\} \qquad (1)$$

$$n \cdot Trans_i \cdot l(x) \le \sum_{e}^{e \in E} Res(e) \qquad (2)$$

$$n \cdot (|S_i| - x) \le \sum_{v}^{v \in V} Res(v) \qquad (3)$$

Sub-chain



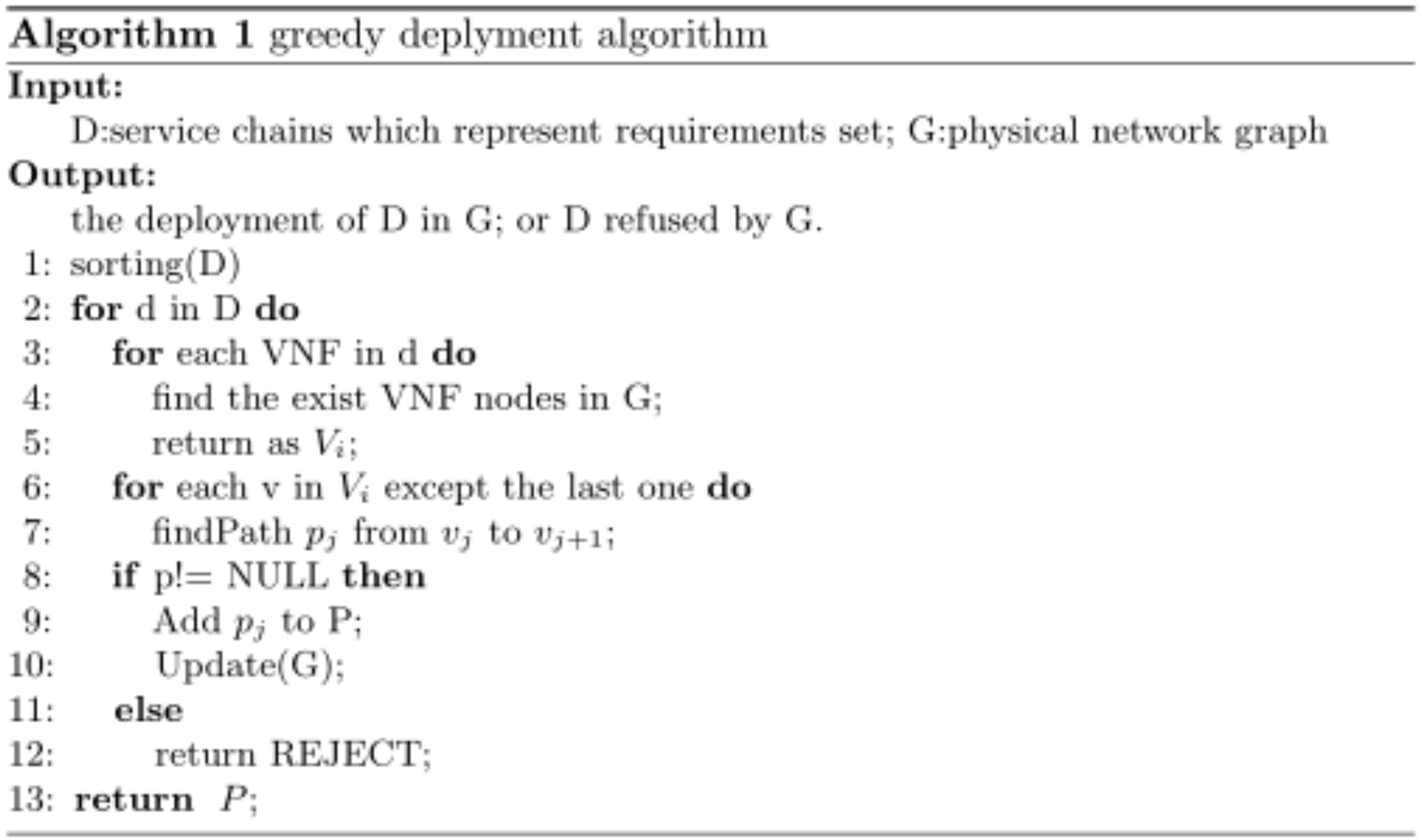
Function c can be reused in node A or B.
Function e can be reused in node C or D.
Else can not be reused.

a-b-c
c-d-e
e-f

A heuristic deployment plan

1. Find the reused function x for an incoming VNF chain.
2. Divide a service chain into (x+1) sub-chains.
3. Greedily find an optimal deployment for each sub-chain, and finally deploy the entire VNF service chain.

Greedy Algorithm



Algorithm 1 greedy deplyment algorithm

Input:

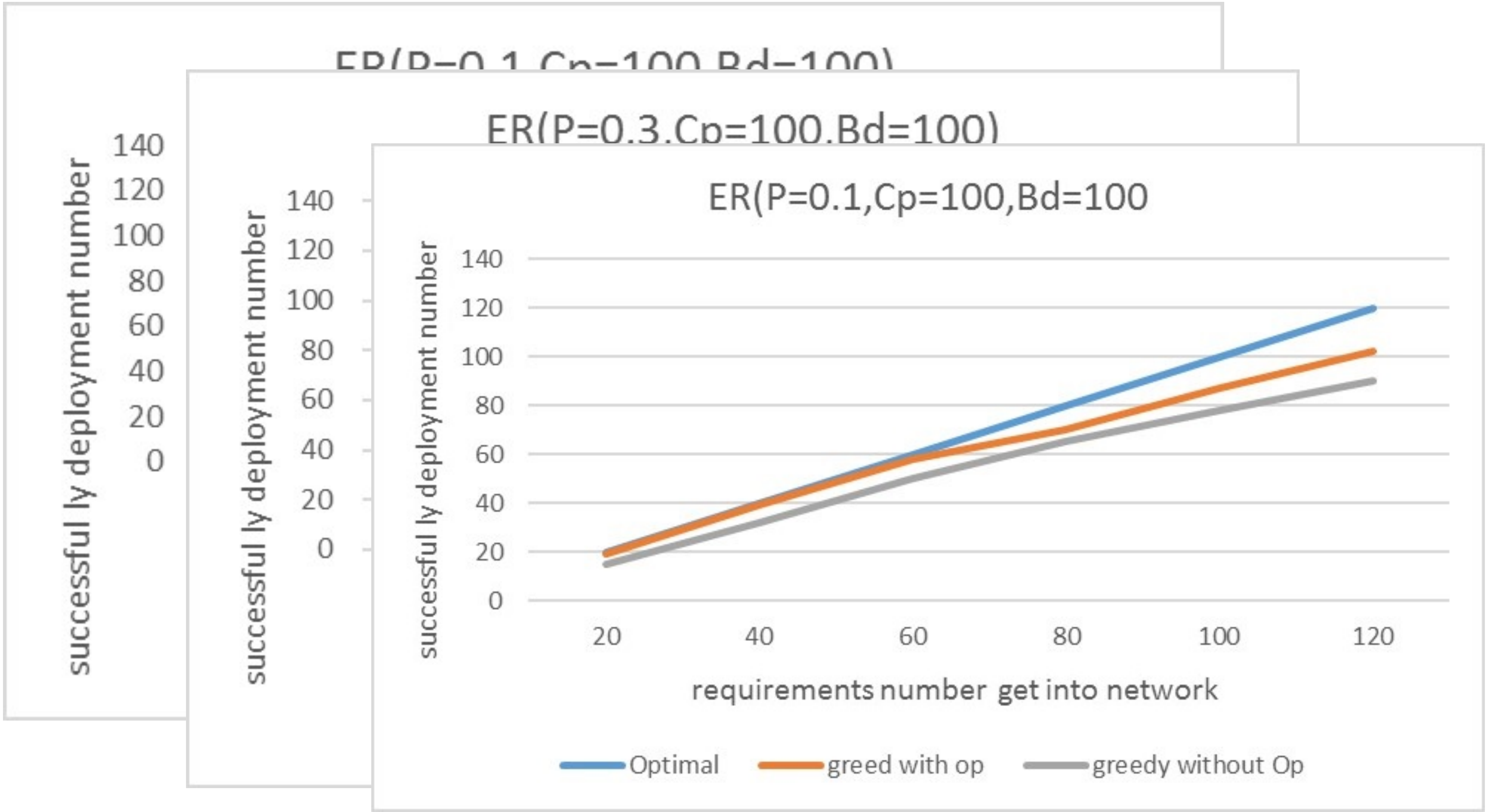
D:service chains which represent requirements set; G:physical network graph

Output:

the deployment of D in G; or D refused by G.

```
sorting(D)
for d in D do
    for each VNF in d do
        find the exist VNF nodes in G;
        return as $V_i$;
    for each v in $V_i$ except the last one do
        findPath $p_j$ from $v_j$ to $v_{j+1}$;
    if p!= NULL then
        Add $p_j$ to P;
        Update(G);
    else
        return REJECT;
return $P$;
```

Simulation justification



Conclusion

- Resource Scheduling for Network Function Virtualization
- Dynamic requirements
- Reused nodes
- Dynamic programming

ensure the stability of the network and also increase the efficiency

Thank you!