# Virtual Network Resource Allocation Considering Dependability Issues

Victor Lira

Orientador: Eduardo Tavares

Centro de Informática U·F·P·E

MoDCS Group

# Introduction

- Internet notably has a vital role in society;
  - Entertainment;
  - Education;
  - Health;
  - So on;

# Internet's Ossification

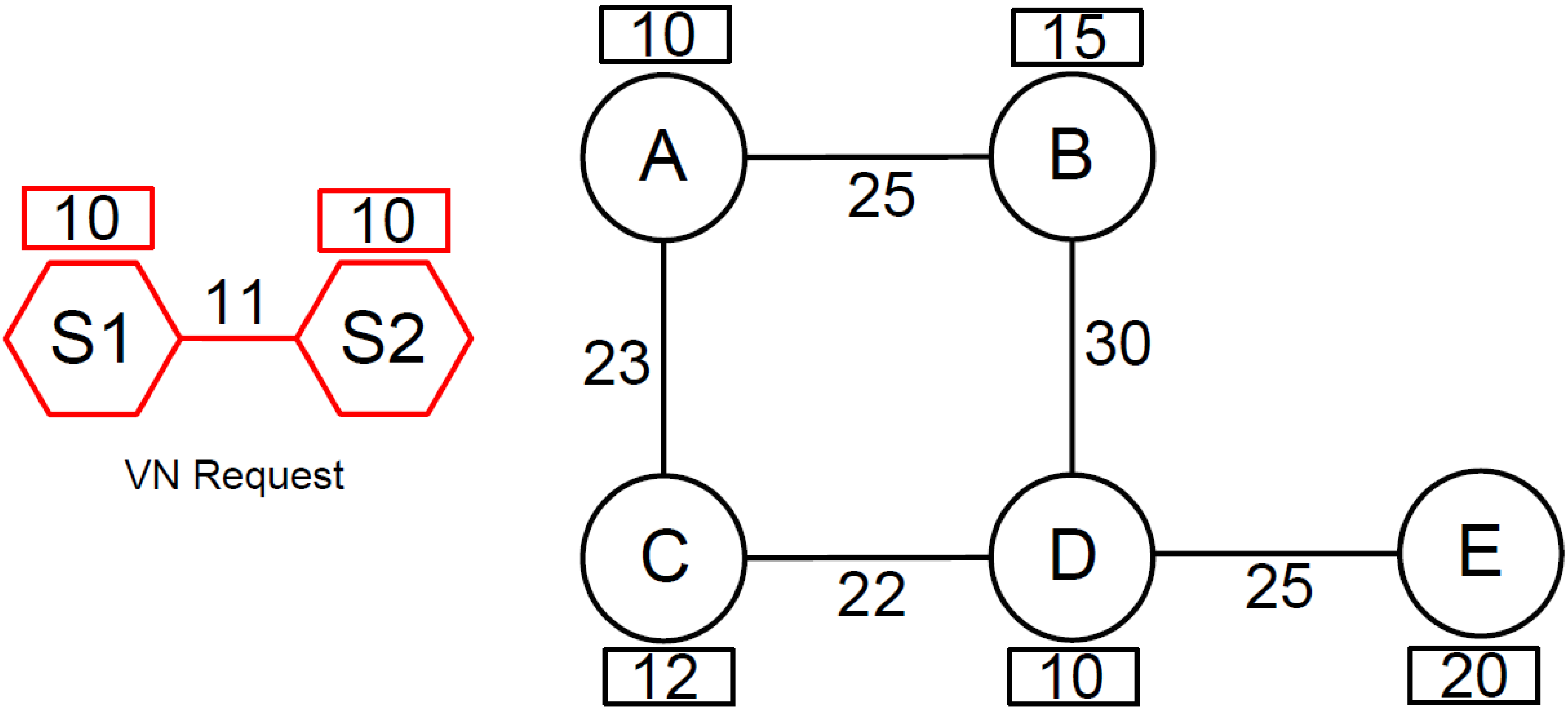
Speed, Capacity, New Applications

Architecture Innovations (e.g., for better mobility support)

# Network Virtualization

- Promising approach to deal with Internet's ossification problem;
- Coexistence of multiple instances of virtual networks on a single shared physical infrastructure;
- Flexibility in the topology, manageability, scalability and traffic isolation;

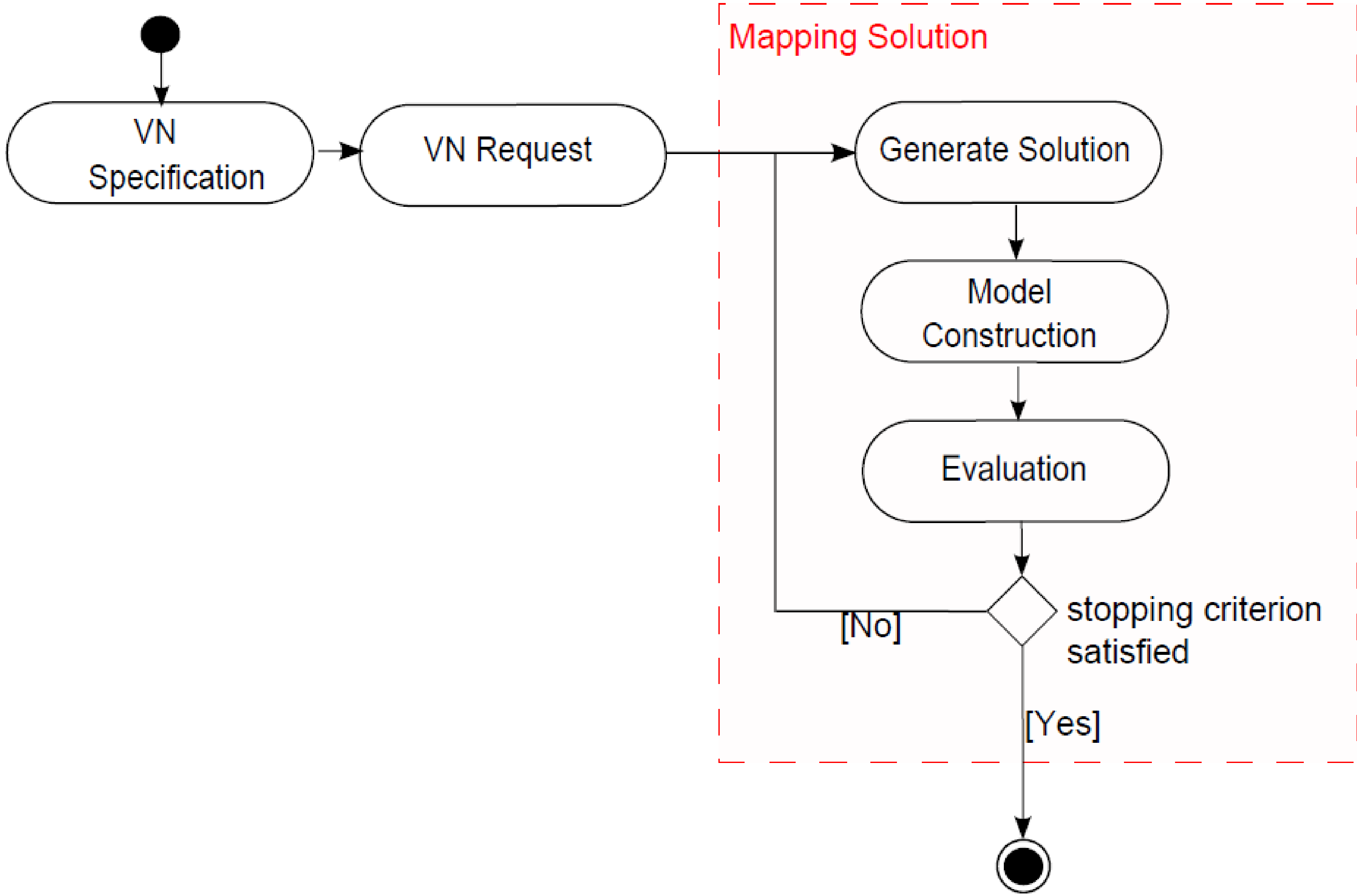
# Network Virtualization



# Dependability

- Ability of a system to deliver a particular service in a reliable way;
- Metric/attribute of interest:
  - Availability;
    - Probability of a system being in a functioning condition. It considers the alternation of operational and nonoperating states;

# Proposed Method



# PROBLEM FORMULATION

# Substrate/Virtual Network

- The physical network is represented by an undirected weighted graph $\mathrm{G}^S = (\mathrm{N}^S, \mathrm{E}^S)$;
  - $n^S \in N^S$ → Nodes;
  - $e^S(i,j) \in E^S$ → Links;

- A VN request is denoted by $G^V = (N^V, E^V)$;
  - $D(G^V)$ → Availability Constraint;

# Substrate Network Resources

- The remaining or available capacity of a physical node, $R_N(n^S), n^S \in N^S$, is defined by:

$$R_N(n^S) = c(n^S) - \sum_{\forall n^V \uparrow n^S} c(n^V)$$

in which $x \uparrow y$ means that the virtual node $x$ is mapped on the physical node $y$

# Substrate Network Resources

- Also, the available bandwidth of a path $P \in P^S$ is given by:

$$R_E(P) = \min_{e^S \in P} R_E(e^S)$$

# Virtual Network Allocation

- For each VN request received, the VNP accepts or rejects the request, according to the available resources and constraints;
- In case of acceptance, a mapping for the VN on the physical network is accomplished, reserving the required network resources;

# Virtual Network Allocation

- VN mapping is split into activities: (i) node mapping and (ii) link mapping.

- Besides, all requests are subject to:

$$Av(G^V) \geq D(G^V)$$

# Node Mapping

- Each virtual node is mapped into a physical node using $M_N : N^V \rightarrow N^S$, so that, $\forall n^V \in N^V$:

$$c(n^V) \leq R_N(M_N(n^V))$$

# Node Mapping

- If redundancy is adopted, an additional mapping $M_{SN} : N^V \to N^S$ is considered, such that, $\forall n^V \in N^V$, $M_{SN}(n^V) \neq M_N(n^V)$ subject to:

$$c(n^V) \leq R_N(M_{SN}(n^V))$$

# Node Mapping

- In addition, considering cold standby redundancy, $\forall n^V \in N^V$:

$$M_{SN}(n^V) \neq M_N(m^V)$$
$$M_N(n^V) = M_N(m^V), iff(n^V = m^V)$$
$$M_{SN}(n^V) = M_{SN}(m^V), iff(n^V = m^V)$$

# Link Mapping

- The mapping of virtual links to physical paths is defined by $M_{ME} : E^V \rightarrow P^S(M_N(m^V), M_N(n^V))$, such that, for any $e^V = (m^V, n^V) \in E^V$:

$$R_E(p) \geq b(e^V), \forall p \in M_{ME}(e^V)$$

Centro de Informática U·F·P·E MoDCS Group

# Link Mapping

- In VN requests with redundancy, three additional virtual links are required due to redundant nodes:

1. Spare-primary*:*
   $M_{SP}: E^V \rightarrow P^S(M_{SN}(m^V), M_N(n^V));$
2. Primary-spare*:*
   $M_{PS} : E^V \rightarrow P^S(M_N(m^V), M_{SN}(n^V));$
3. Spare-spare:
   $M_{SS} : E^V \rightarrow P^S(M_{SN}(m^V), M_{SN}(n^V));$

# Link Mapping

- They are mappings from virtual links to physical paths, such that, for any $e^V = (m^V, n^V) \in E^V$,

$$R_E(p) \;\geq b(e^V)\,, \forall p \in M_{SP}(e^V)$$
$$R_E(p) \;\geq b(e^V)\,, \forall p \in M_{PS}(e^V)$$
$$R_E(p) \;\geq b(e^V)\,, \forall p \in M_{SS}(e^V)$$

Centro de Informática U·F·P·E MoDCS Group

# Objective

- Allocating VN requests to meet specified constraints (e.g., availability), minimizing the cost resulting from allocations:

$$\sum_{e^V \in E^V} \sum_{e^S \in E^S} f_{e^S}^{e^V} + \sum_{n^V \in N^V} c(n^V) * x$$

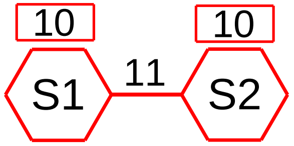
in which $f_{e^S}^{e^V}$ represents the total bandwidth allocated on link $e^S$ to the virtual link $e^V$.

$x$ is an integer variable , which is equal to '2' whenever redundancy is considered on VN request. Otherwise, the value is equal to '1'.

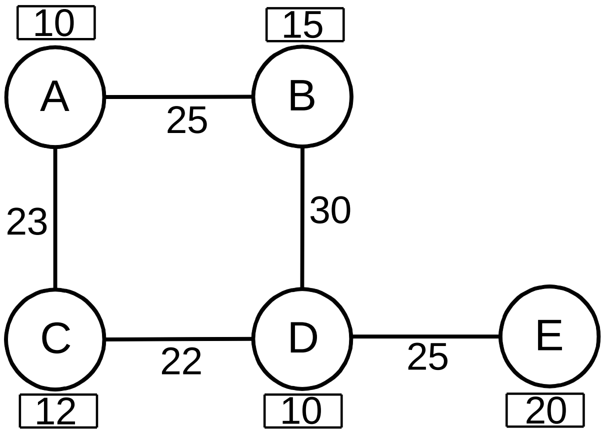
# DEPENDABILITY MODELING

# No redundancy

- $M_N(S1) = A;$
- $M_N(S2) = B;$
- $M_{ME}(S1, S2) = (A, B);$

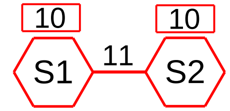


VN Request

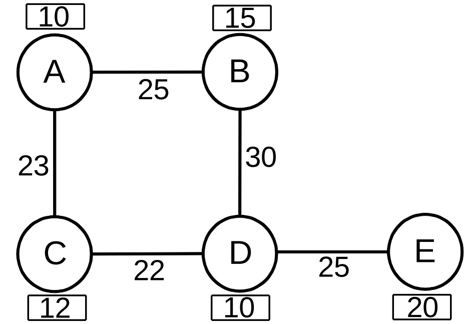


Substrate Network Topology

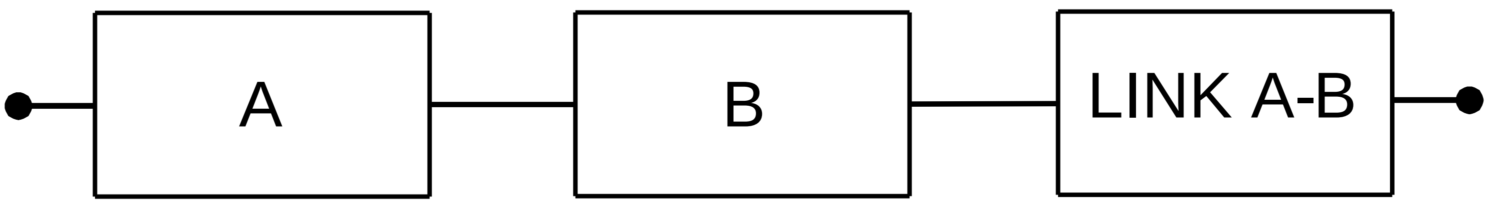
# No redundancy




VN Request

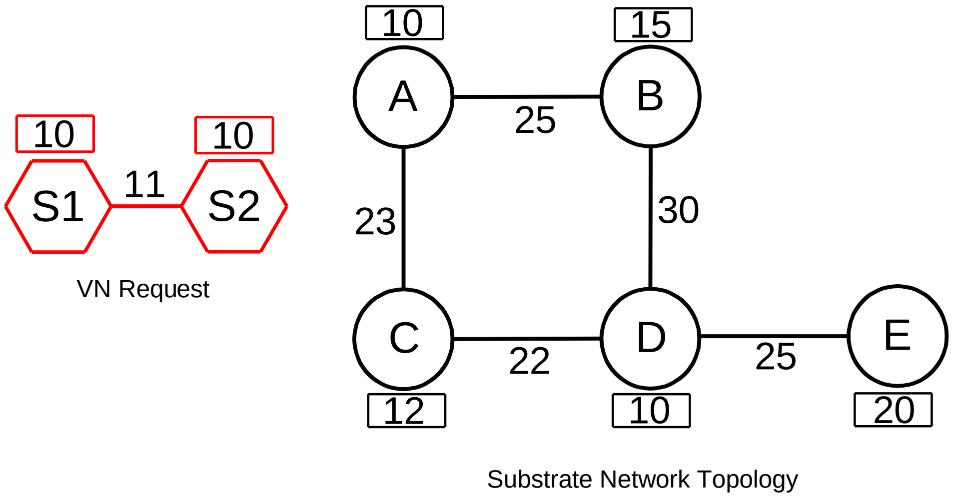



Substrate Network Topology

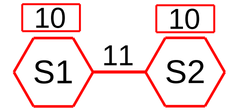


# Hot Standby

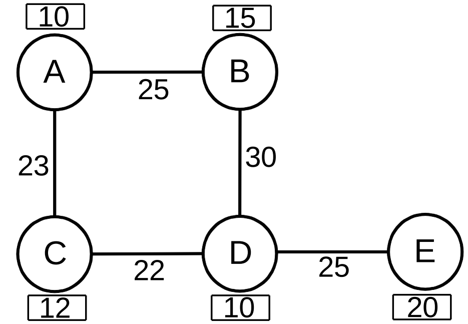
- $M_N(S1) = C;$
- $M_{SN}(S1) = E;$
- $M_N(S2) = D;$
- $M_{SN}(S2) = E;$
- $M_{ME}(S1, S2) = \{(C, D)\};$
- …



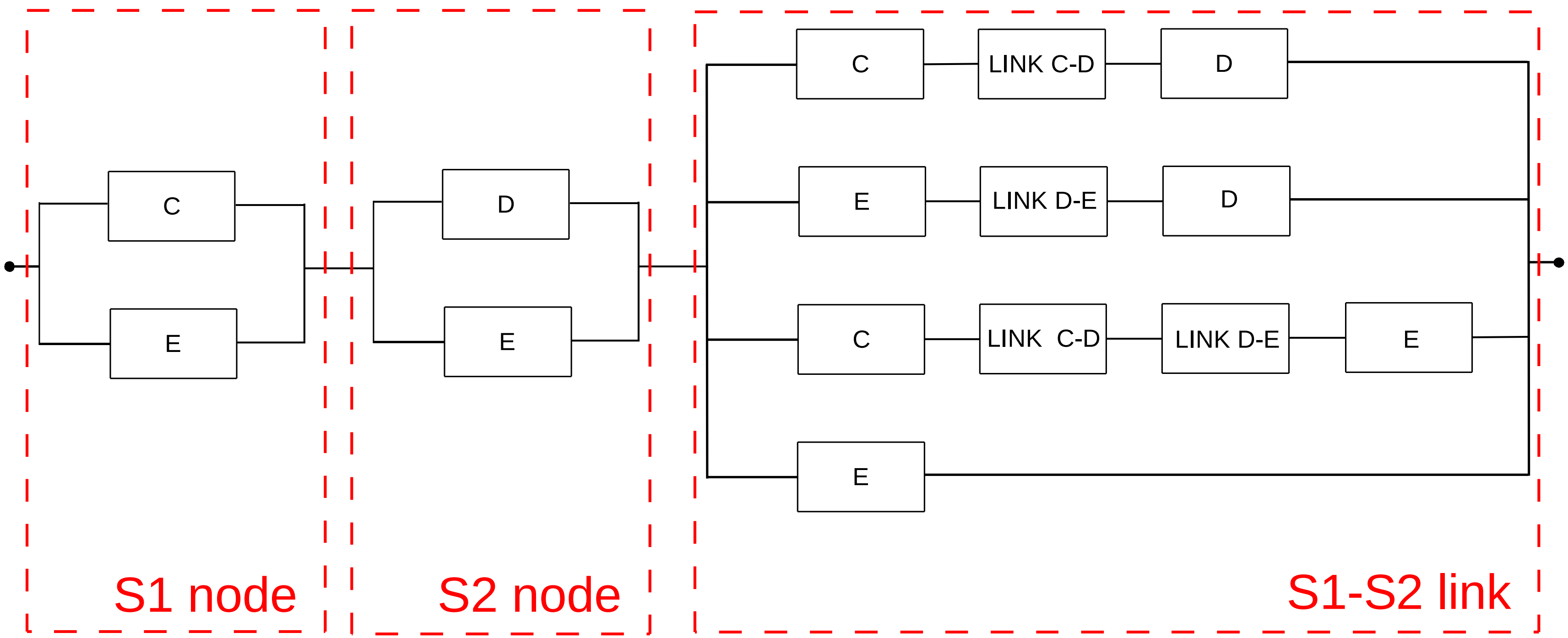
# Hot Standby




VN Request

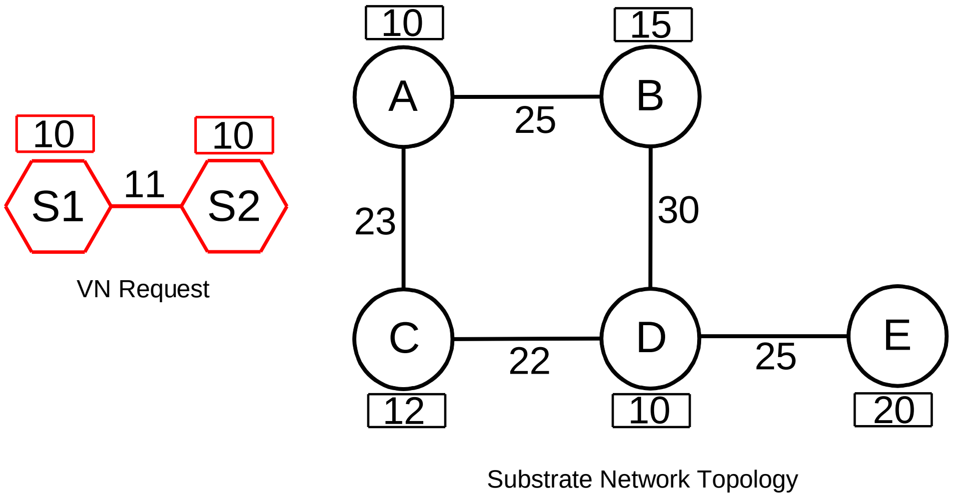



Substrate Network Topology



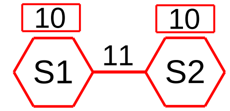
# Cold Standby

- $M_N(S1) = A;$
- $M_{SN}(S1) = B;$
- $M_N(S2) = C;$
- $M_{SN}(S2) = D;$
- $M_{ME}(S1, S2) = \{(A, C)\};$
- …

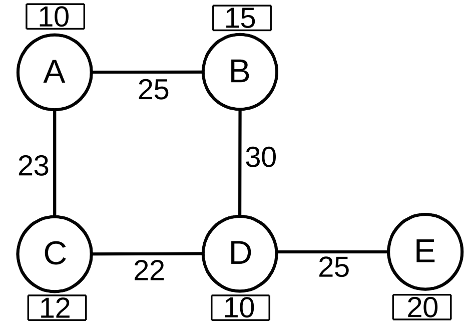



Centro de Informática U·F·P·E MoDCS Group

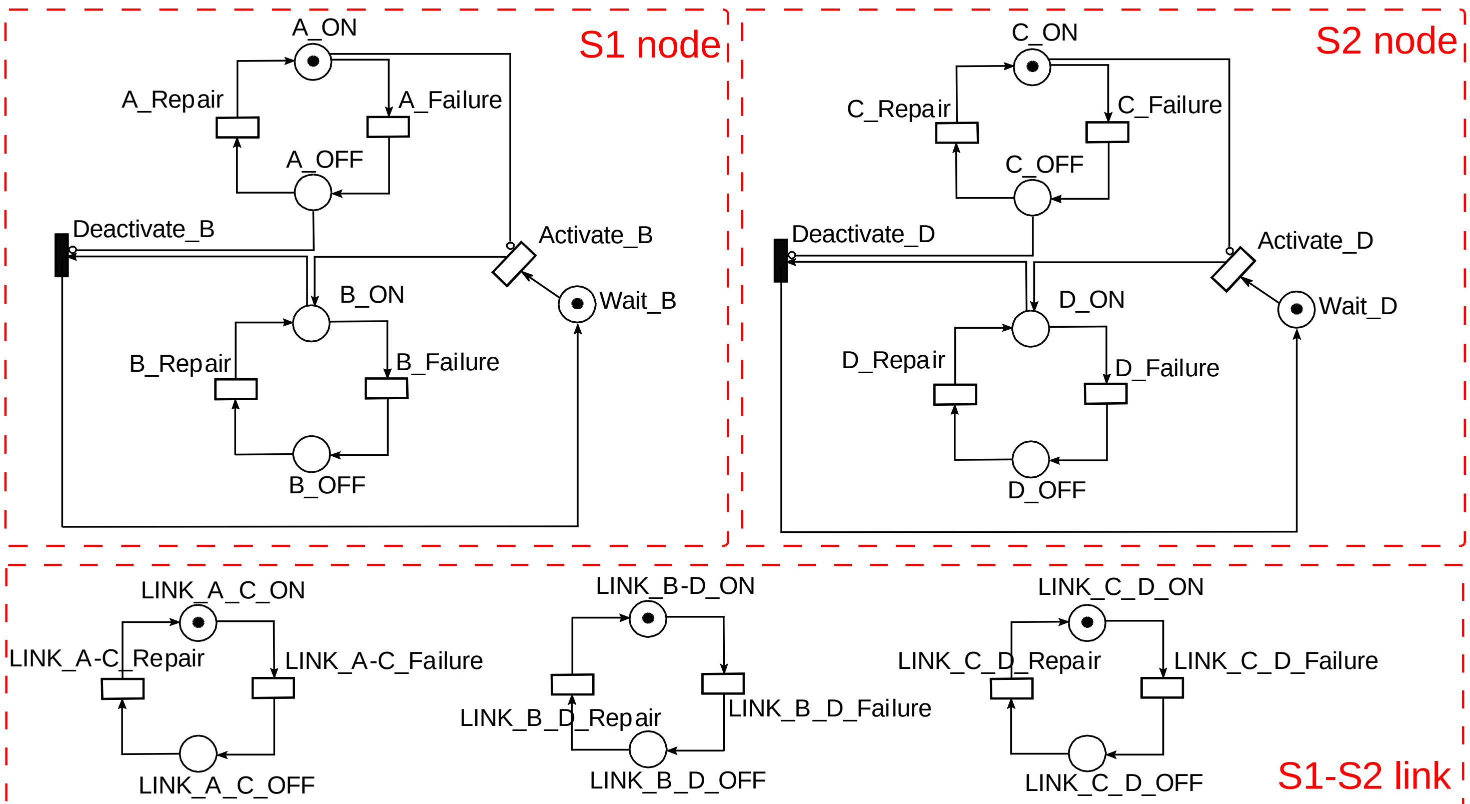
# Cold Standby



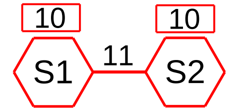

VN Request



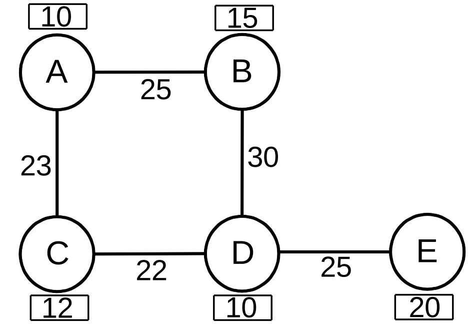

Substrate Network Topology



# Cold Standby



VN Request



Substrate Network Topology

```
P{(
  ((#A_ON + #B_ON>0))                                                    S1 node
  AND
  (
     ((#A_ON > 0)AND(#LINK_A_C_ON > 0)AND(#C_ON > 0))
     OR
     ((#B_ON > 0)AND(#LINK_B_D_ON > 0)AND(#LINK_C_D_ON > 0)AND(#C_ON > 0))
     OR
     ((#A_ON > 0)AND(#LINK_A_C_ON > 0)AND(#LINK_C_D_ON > 0)AND(#D_ON > 0))
     OR
     ((#B_ON > 0)AND(#LINK_B_D_ON > 0)AND(#D_ON > 0))
   )                                                                     S1-S2 link
   AND
   ((#C_ON + #D_ON>0))                                                   S2 node
)}
```

# GRASP FOR VIRTUALIZED NETOWRK ALLOCATION

# GRASP

- GRASP (Greedy Randomized Adaptive Search Procedure);
- Two phases:
  - Construction;
  - Local search;

**procedure** GRASP($G^S = (N^S, E^S)$, $G^V = (N^V, E^V)$)
    Best_Solution ← ∅;
    **while** stopping criterions not satisfied **do**
        Solution ← GreedyRandomized ($G^S$, $G^V$);
        **if** Solution ≠ ∅ **then**
            Solution ← LocalSearch(Solution, $G^S$, $G^V$);
            **if** $f$(Solution) < $f$(Best_Solution) **then**
                Best_Solution ← Solution;
            **end if**
        **end if**
    **end while**
    **if** Best_Solution = ∅ **then**
        **return** VN request cannot be satisfied;
    **end if**
    **return** Best_Solution;
**end procedure**

# GRASP – Construction Phase

**procedure** GreedyRandomized ($G^S = (N^S, E^S), G^V = (N^V, E^V)$)<br>
**for all** $n^V \in N^V$ **do**<br>
    Randomly select a feasible node $M_N(n^V)$ from the $N^S$;<br>
    **if** consider redundancy<br>
        Randomly select a feasible node $M_{SN}(n^V)$ from the $N^S$;<br>
    **end if**<br>
**end for**<br>
**for all** $e^V(i,j) \in E^V$ **do**<br>
    Solve shortest feasible path $M_{ME}(e^V)$ from $M_N(i)$ to $M_N(j)$;<br>
    **if** consider redundancy<br>
        Solve shortest feasible path $M_{SP}(e^V)$ from $M_{SN}(i)$ to $M_N(j)$;<br>
        Solve shortest feasible path $M_{PS}(e^V)$ from $M_N(i)$ to $M_{SN}(j)$;<br>
        Solve shortest feasible path $M_{SS}(e^V)$ from $M_{SN}(i)$ to $M_{SN}(j)$;<br>
    **end if**<br>
**end for**<br>
Solution $\leftarrow (M_N, M_{SN}, M_{ME}, M_{SP}, M_{PS}, M_{SS})$;<br>
Calculate Solution cost;<br>
Calculate Solution dependability;<br>
**if** Solution is not feasible **then**<br>
    **return** $\emptyset$**;**<br>
**end if**<br>
**return** Solution;<br>
**end procedure**

MoDCS Group

# GRASP – Local Search

```
procedure LocalSearch(Solution, $G^S = (N^S, E^S)$, $G^V = (N^V, E^V)$)
for all $M_N(n^V) \in N^V$ do
    Neighbors ← Select all neighbors of $M_N(n^V)$;
    for all $n^S \in$ Neighboors do
        Local_Solution ← Build a feasible solution by replacing $M_N(n^V)$ by $n^S$;
        if $f$(Local_Solution) < $f$(Solution) then
            return LocalSearch(Local_Solution, $G^S$, $G^V$);
        end if
    end for
    if has redundancy
        Spare_Neighbors ← Select all neighbors of $M_{SN}(n^V)$;
        for all $n^S \in$ Spare_Neighbors do
            Local_Solution ← Build a feasible solution by replacing $M_{SN}(n^V)$ by $n^S$;
            if $f$(Local_Solution) < $f$(Solution) then
                return LocalSearch(Local_Solution, $G^S$, $G^V$);
            end if
        end for
    end if
end for
return Solution;
end procedure
```

# EXPERIMENTAL RESULTS

# Experiment Settings

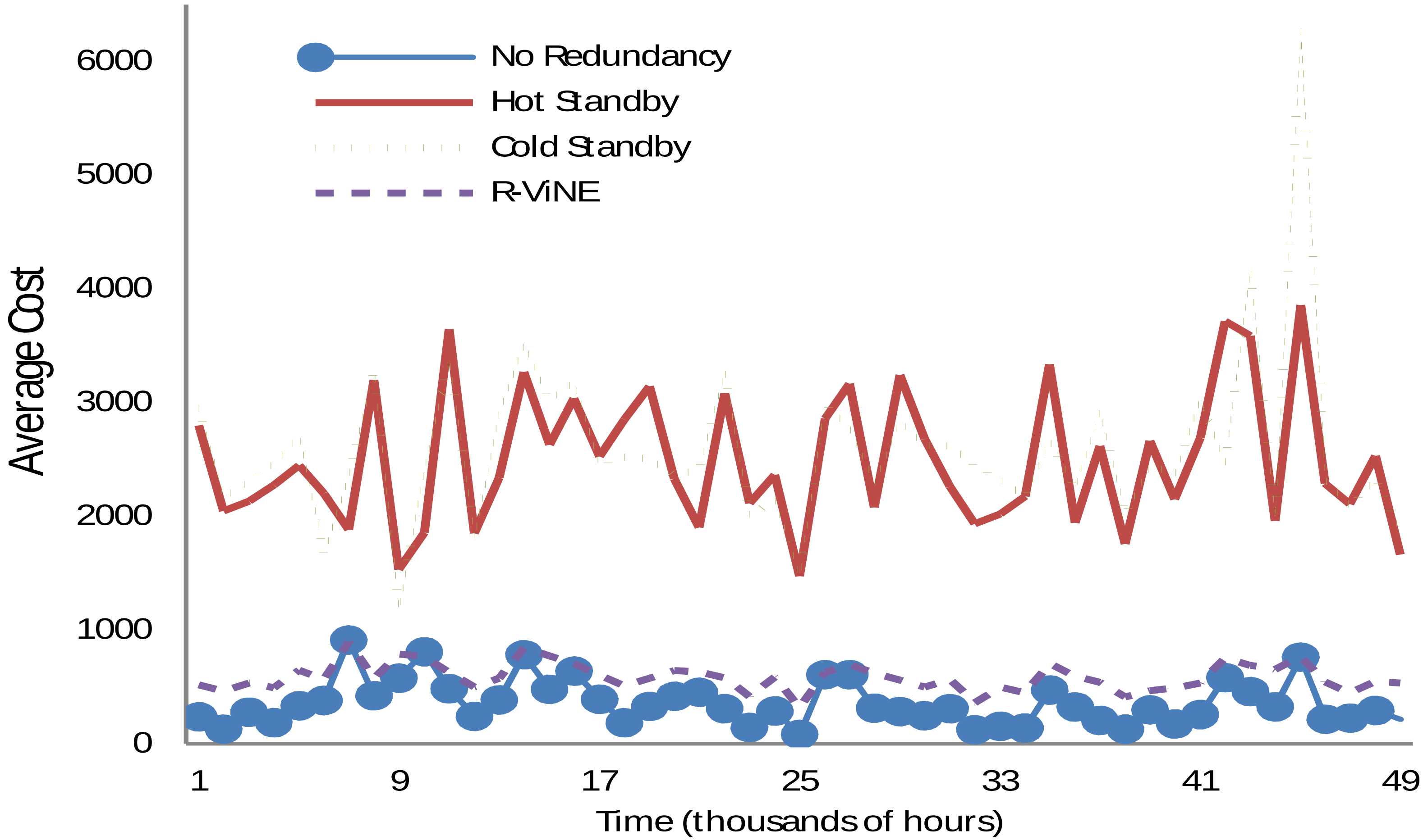
- GT-ITM tool to generate the physical network topology;
- Substrate network:
  - 50 nodes randomly conected with probability 0.5;
  - Nodes capacities and link bandwidths are real numbers uniformly distributed between 50 and 100;

# Experiment Settings

- 800 VN requests are considered over a period of 50,000 hours;
- 0.9 (90%) is the availability constraint for each VN request;

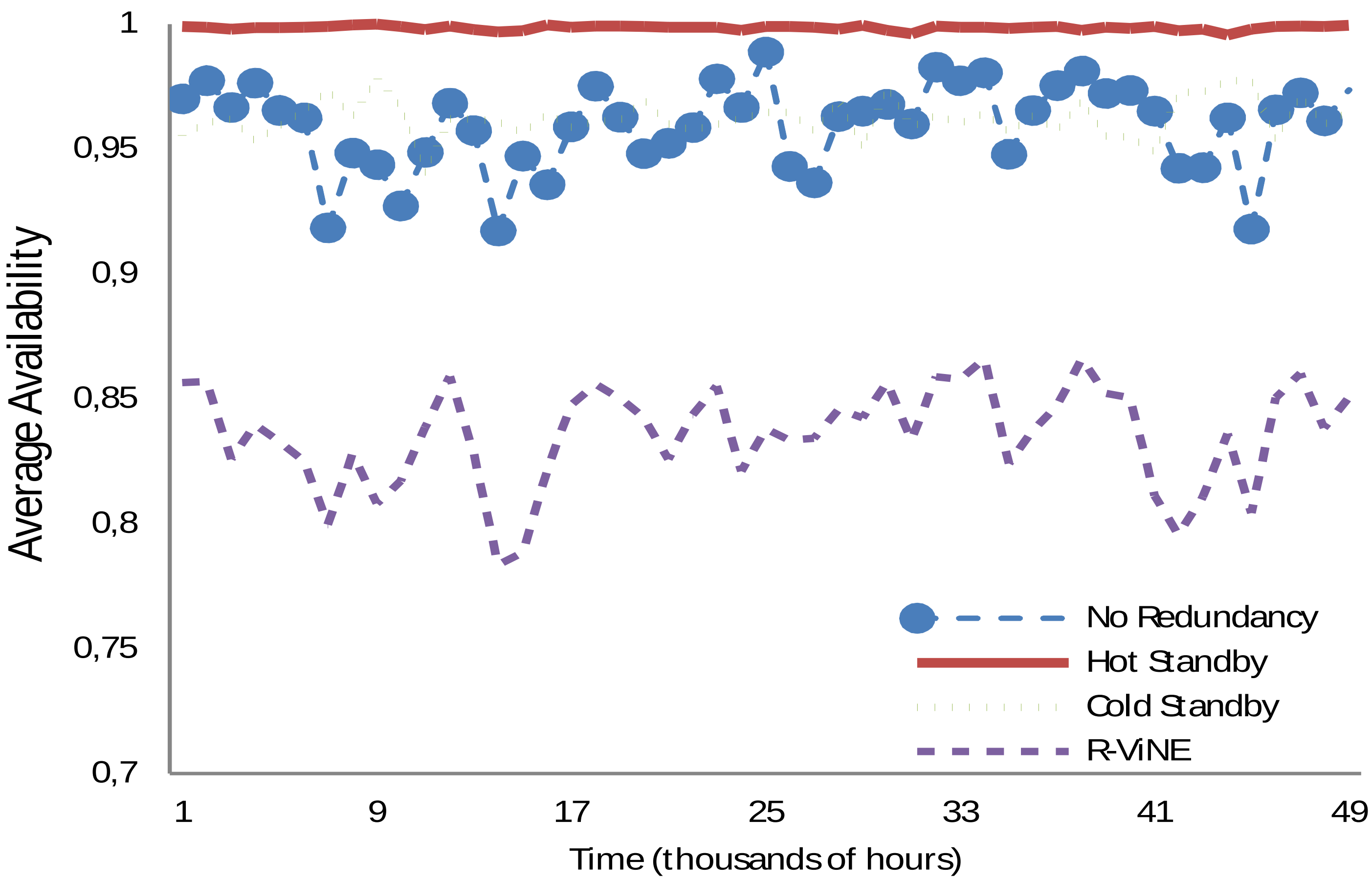
Centro de Informática U·F·P·E MoDCS Group

# Results - Cost

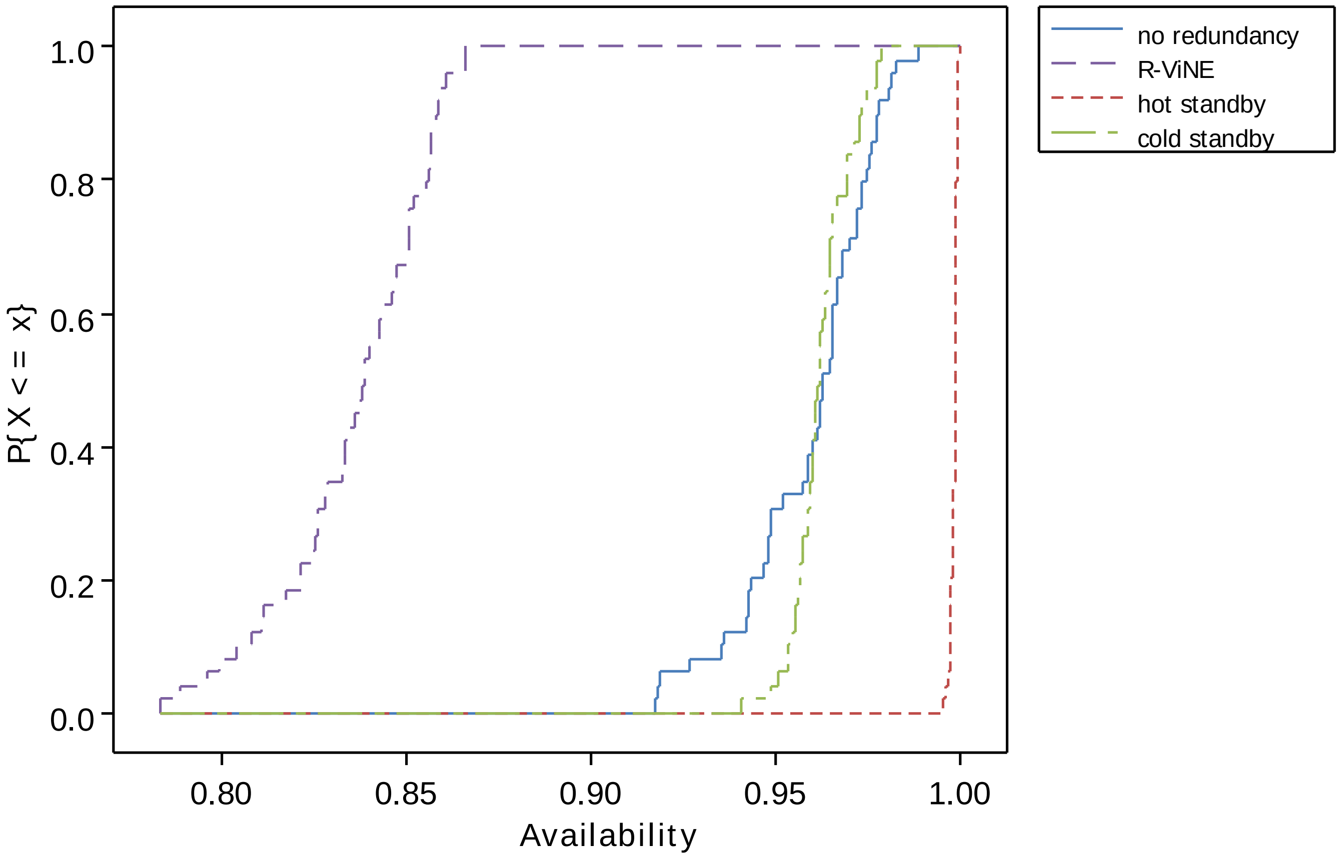



Centro de Informática UFPE
MoDCS Group

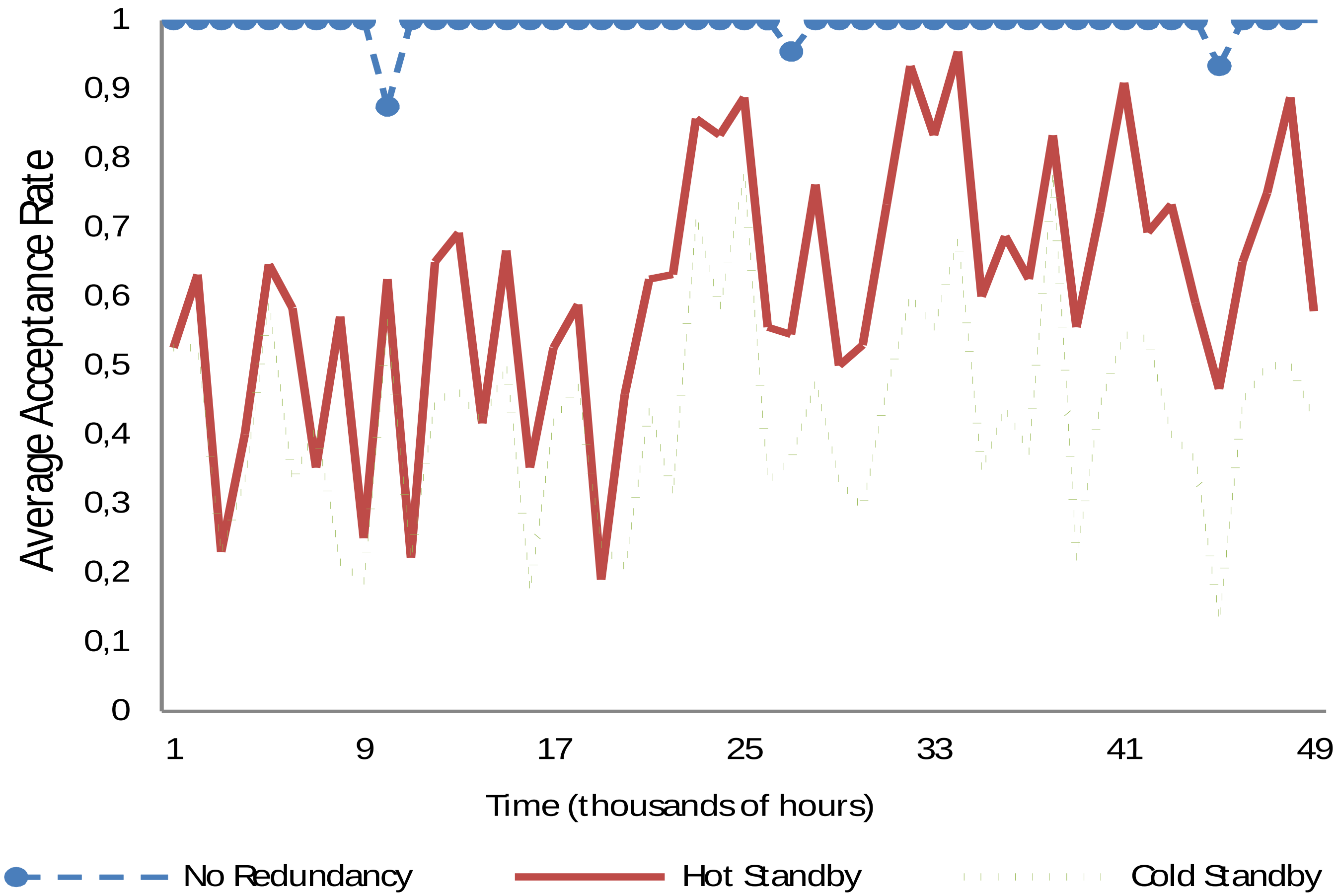
# Results - Availability



# Results – Availability ECDF



# Results – Acceptance Rate



# Conclusion

- Network Virtualization has received particular attention from the scientific community, as several VNs can coexist in the same physical network;
- Many algorithms have been proposed to allocate VNs considering performance metrics. However, dependability is usually neglected.

# Conclusion

- This work proposes a GRASP-based algorithm for allocating virtual networks taking into account dependability issues;

# Thanks!

Centro de Informática U·F·P·E

MoDCS Group