Server Consolidation in Clouds through Gossiping

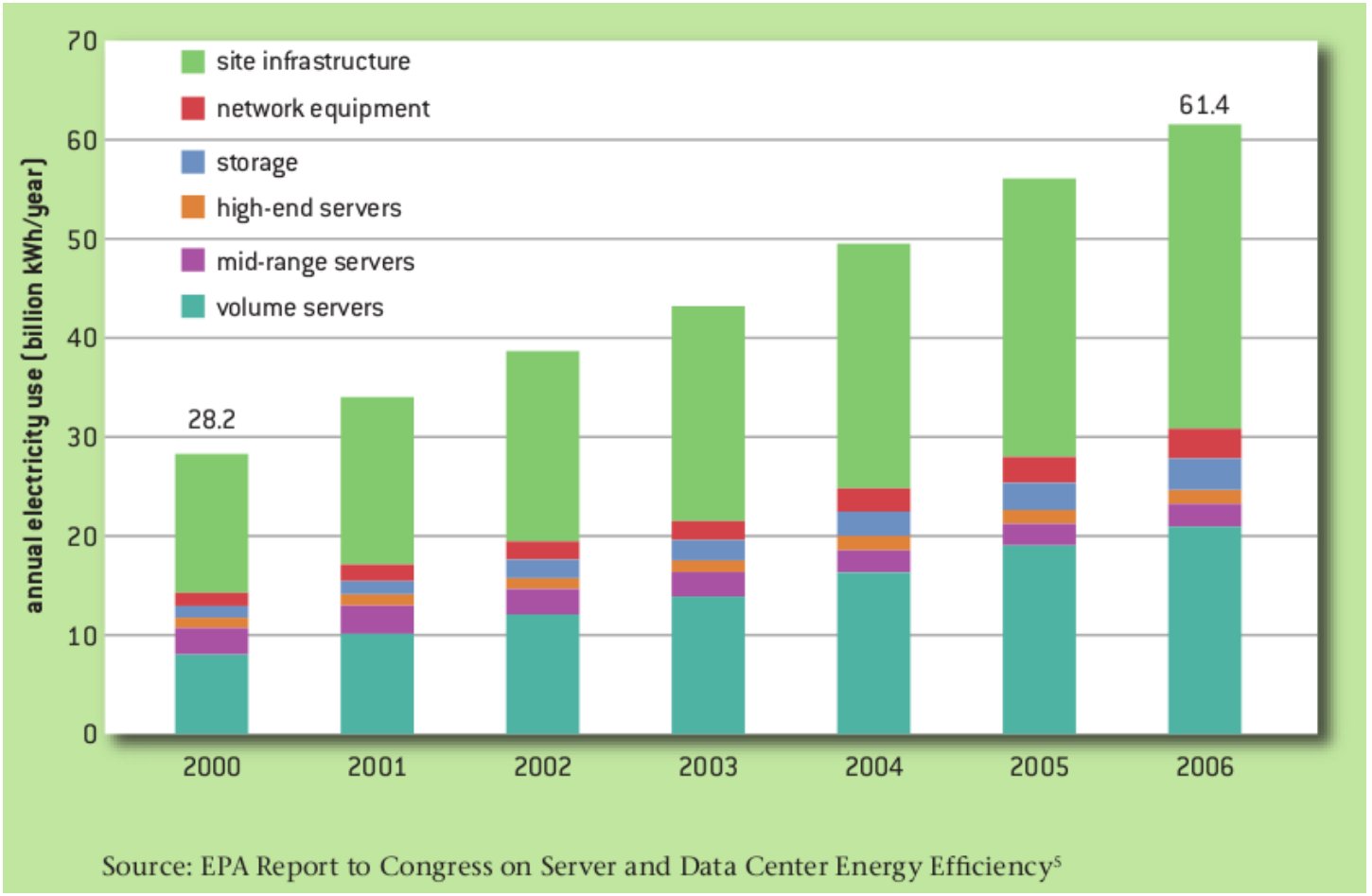
Moreno Marzolla Ozalp Babaoglu Fabio Panzieri

Università di Bologna, Dip. di Scienze dell'Informazione
{marzolla, babaoglu, panzieri}@cs.unibo.it
http://www.moreno.marzolla.name/

Introduction and Motivations

- The success of the Cloud Computing paradigm has resulted in the creation of large datacenters
- The electricity bill accounts for a substantial fraction of the total operational costs, and servers are responsible for a substantial fraction of the electrical power consumed

Electricity use breakdown

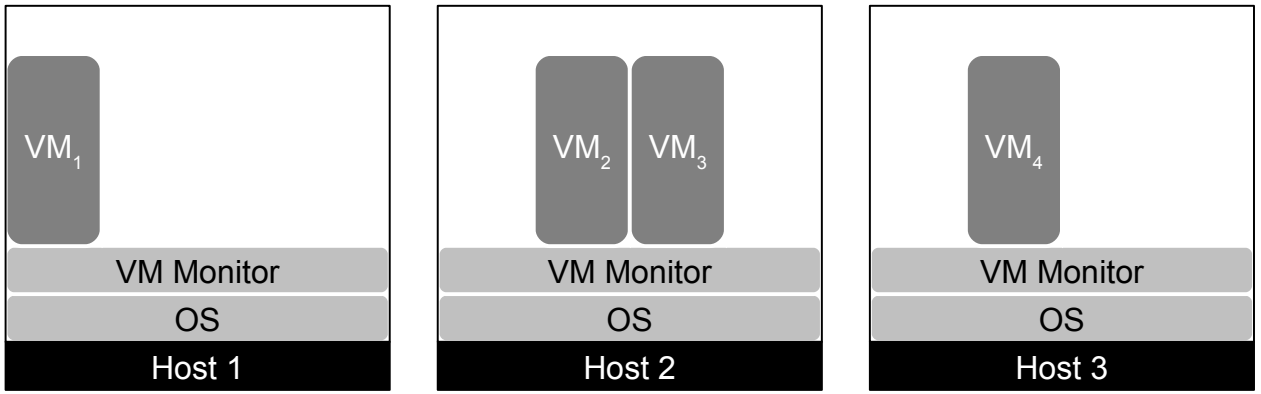


Source: *Toward energy efficient computing*, ACM Queue, http://queue.acm.org/detail.cfm?id=1730791

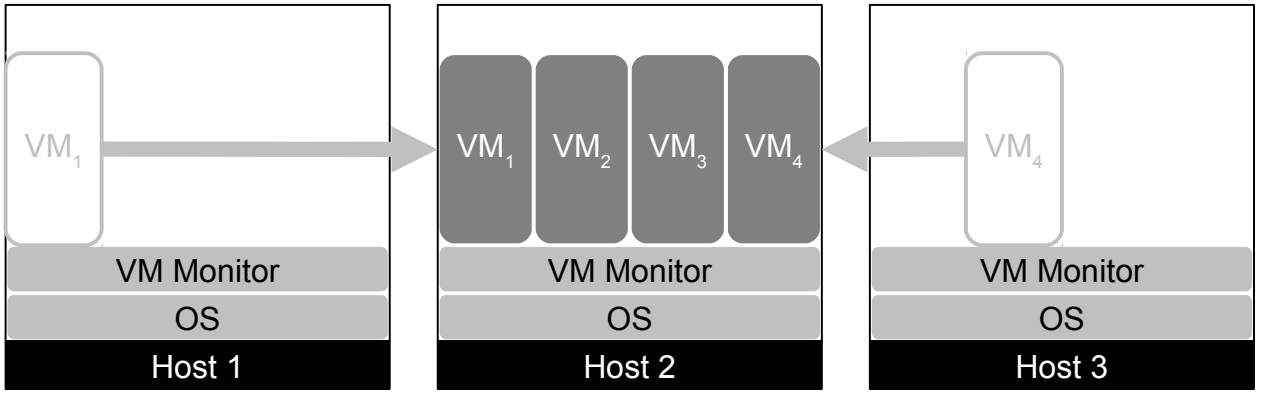
Virtualization

- Main feature of a Cloud system
  - *Dynamic scalability* (pay-as-you-go economic model)
  - *Virtualization* of resources
- Assumptions
  - Physical resources (servers) are multi-core/multi-processor machines; each server can support up to *C* VM instances
  - Cloud users request a set of Virtual Machine (VM) instances
  - Users release instances when no longer needed
  - VM instances can be acquired or released at any time
  - The VM monitor supports *live migration* of VMs
- Our goal
  - Minimize energy consumption by *consolidating VMs*

VM Consolidation



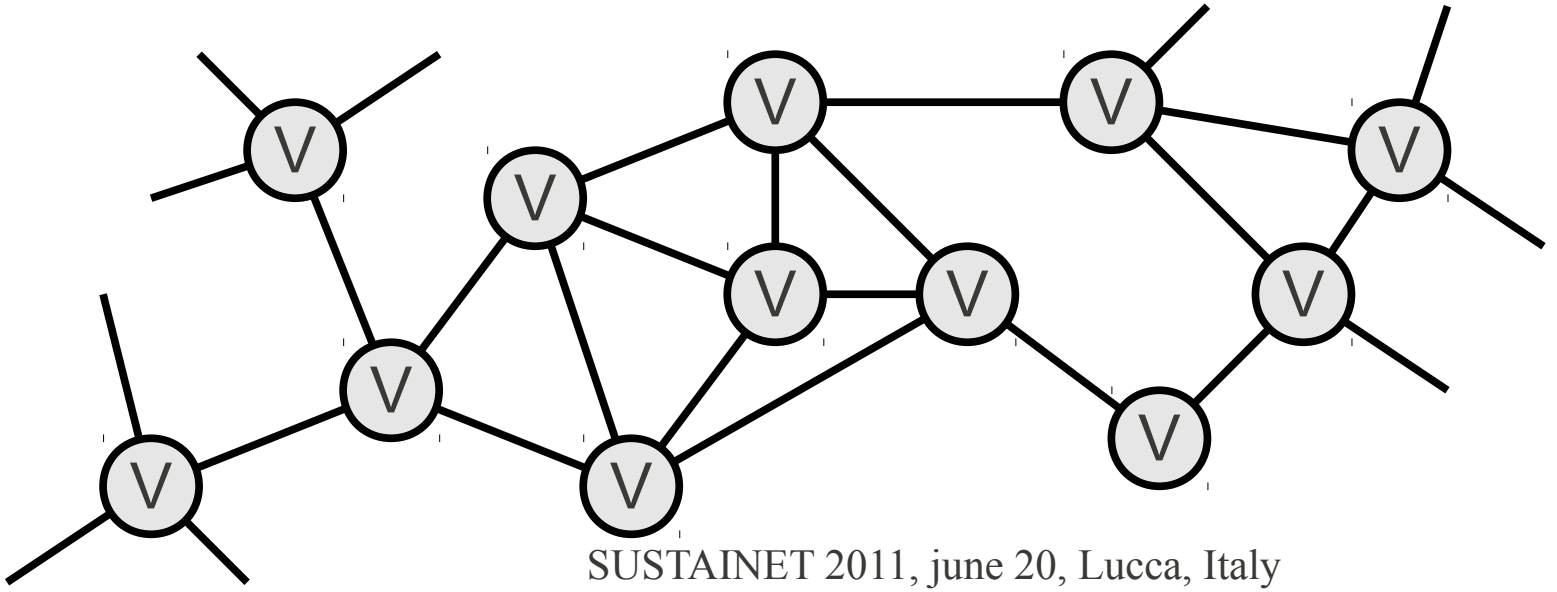
(a) Before consolidation



(b) After consolidation of VM_1 and VM_4 to host 2

VM consolidation through gossiping

- Each server hosts the V-MAN daemon
- Daemons maintain an overlay network such that each daemon is connected to at most *K* other nodes
- Daemons exchange messages only with neighbors
- The overlay is maintained with the *Newscast* algorithm

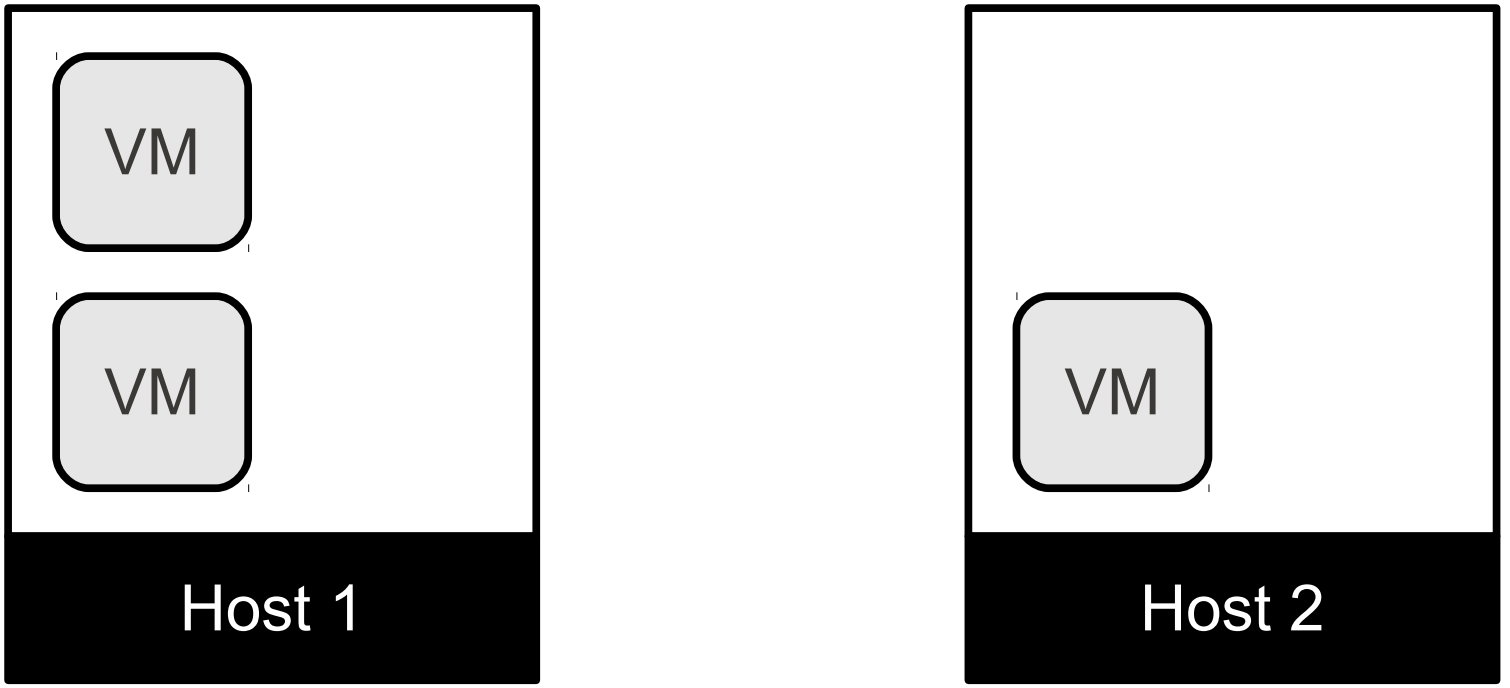


Idea

- Node *I* interacts with each neighbor *j*
- If *I* has fewer VMs than *j*
  - VMs are migrated from *I* to *j*
- If *I* has more VMs than *j*
  - VMs are migrated from *j* to *I*

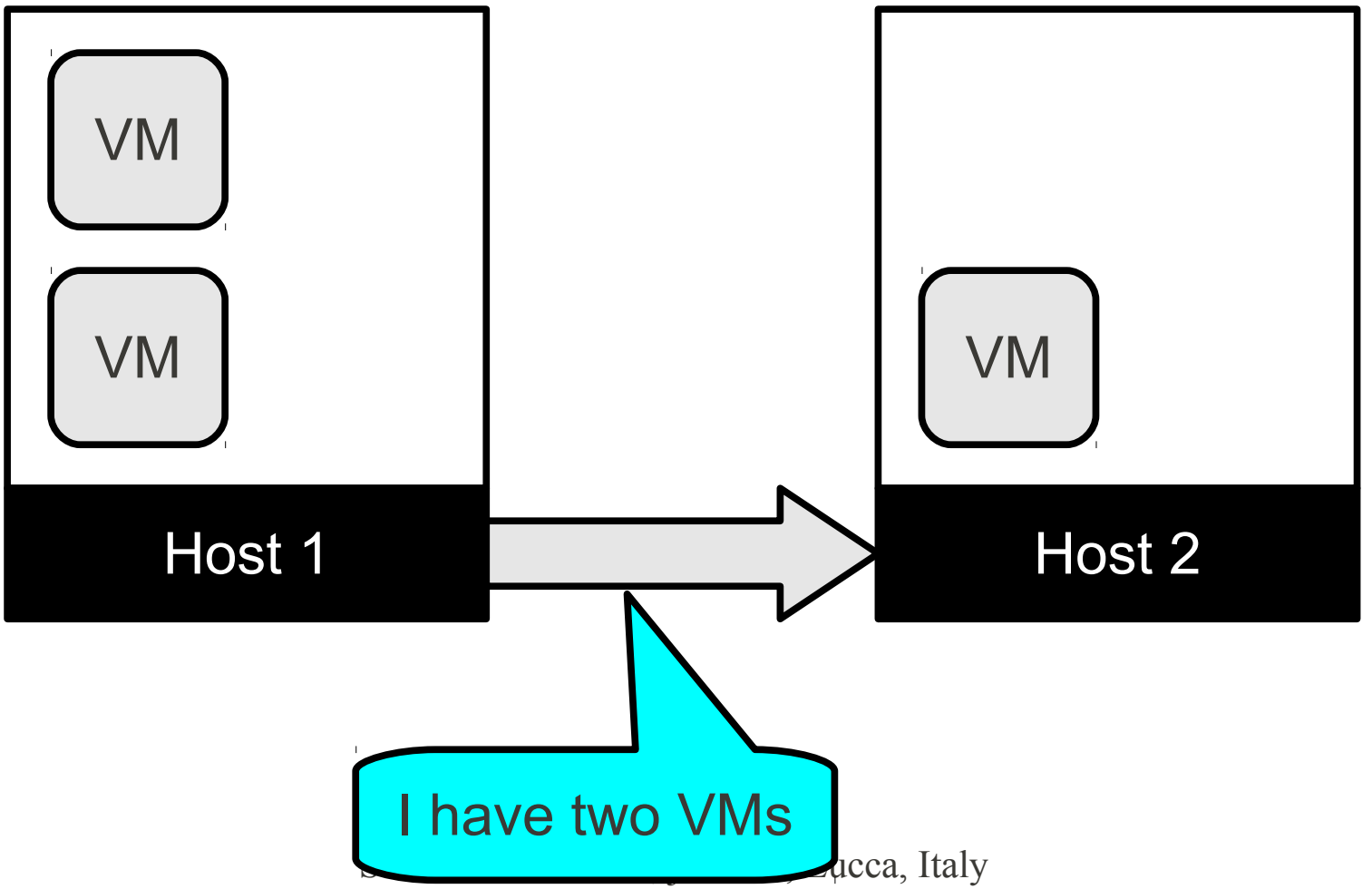
Example (1)

- Capacity = 4



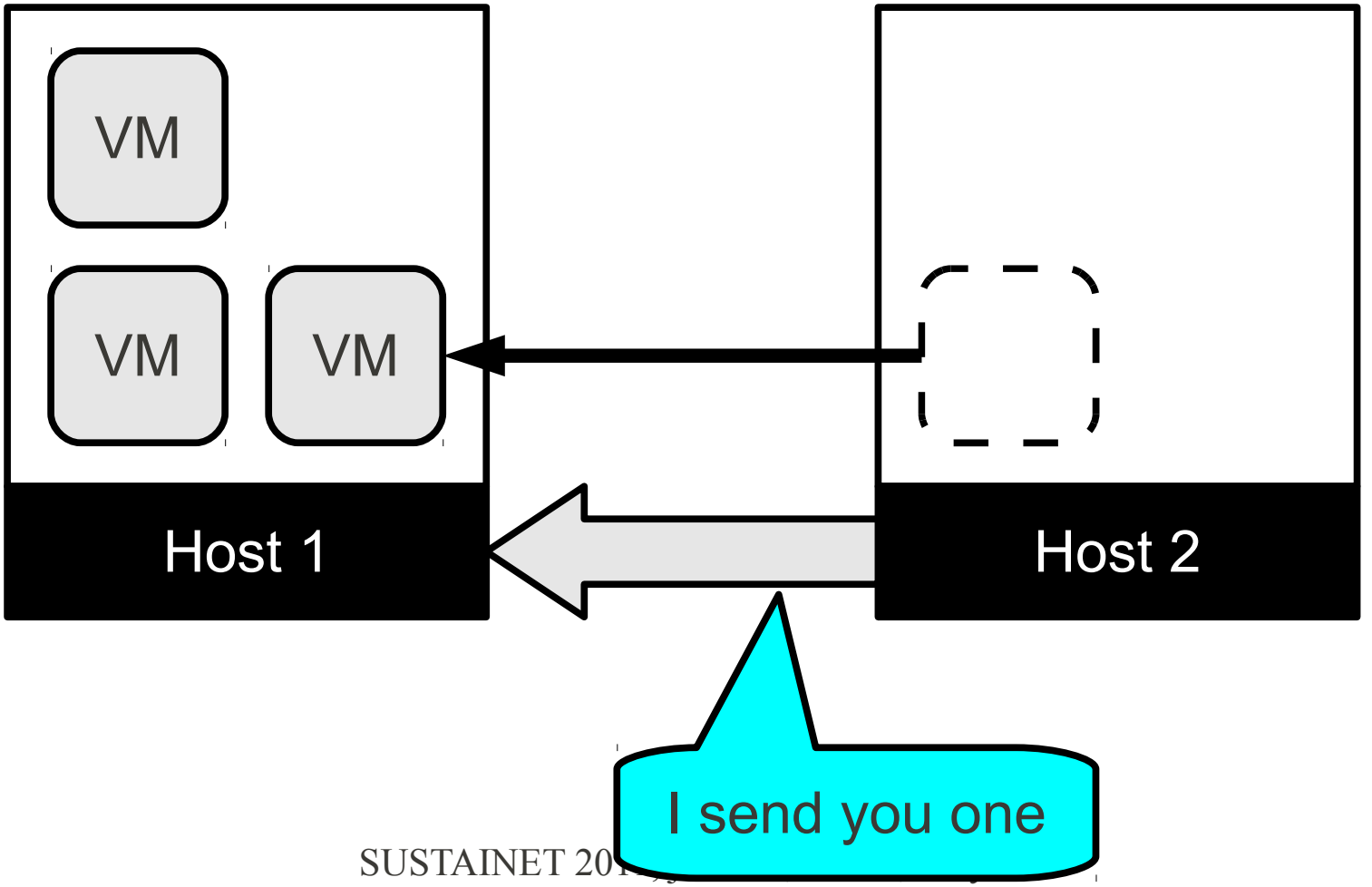
Example (1)

- Capacity = 4



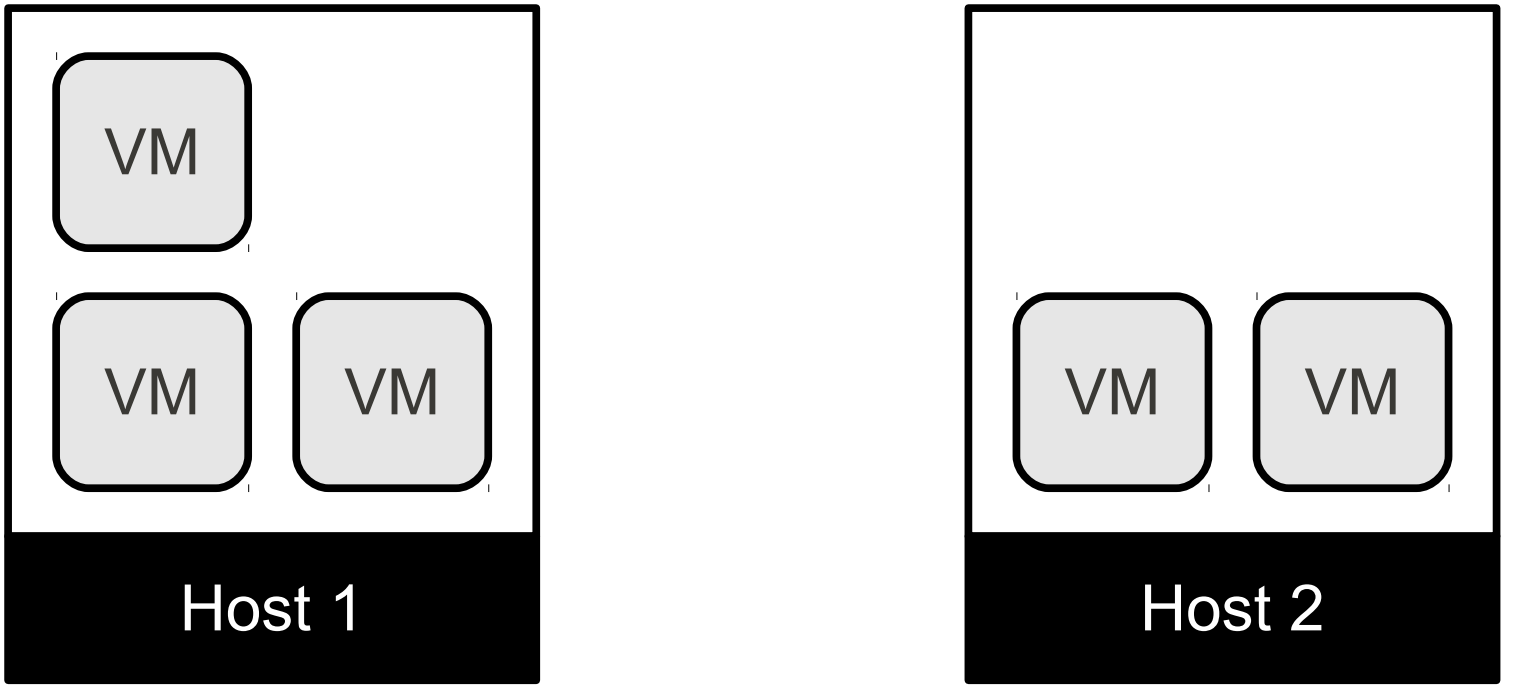
Example (1)

- Capacity = 4



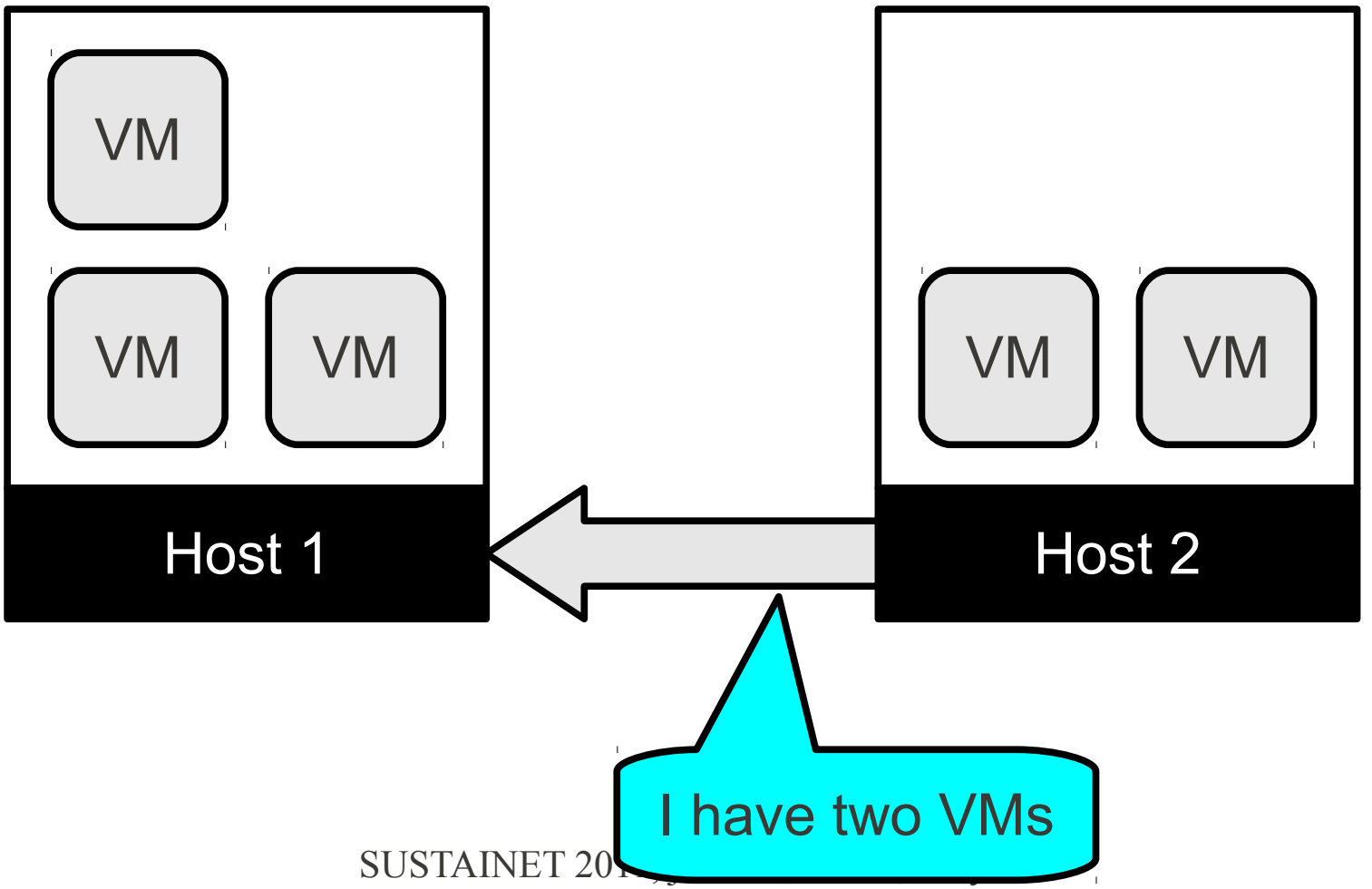
Example (2)

- Capacity = 4



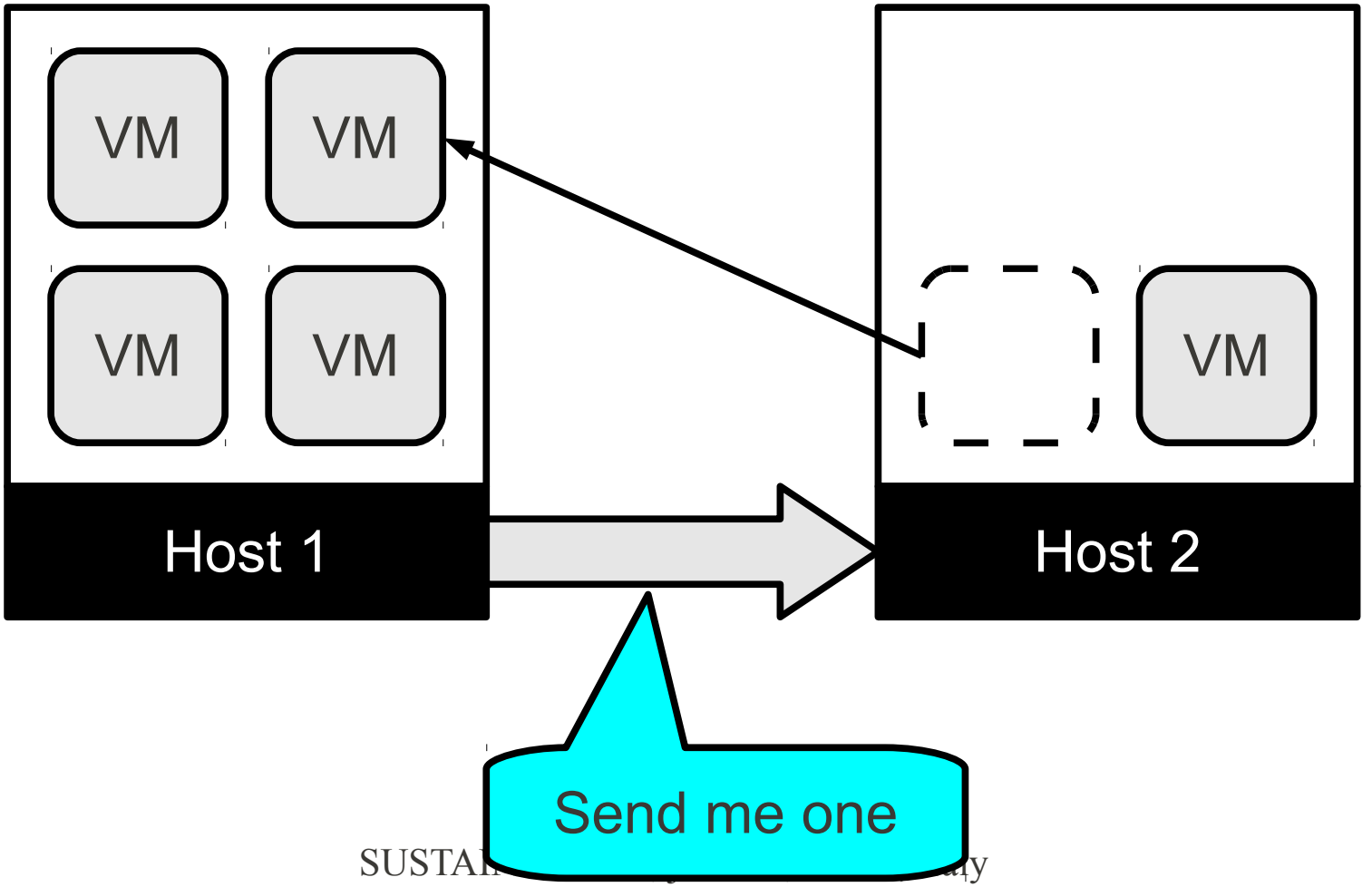
Example (2)

- Capacity = 4



Example (2)

- Capacity = 4



V-MAN

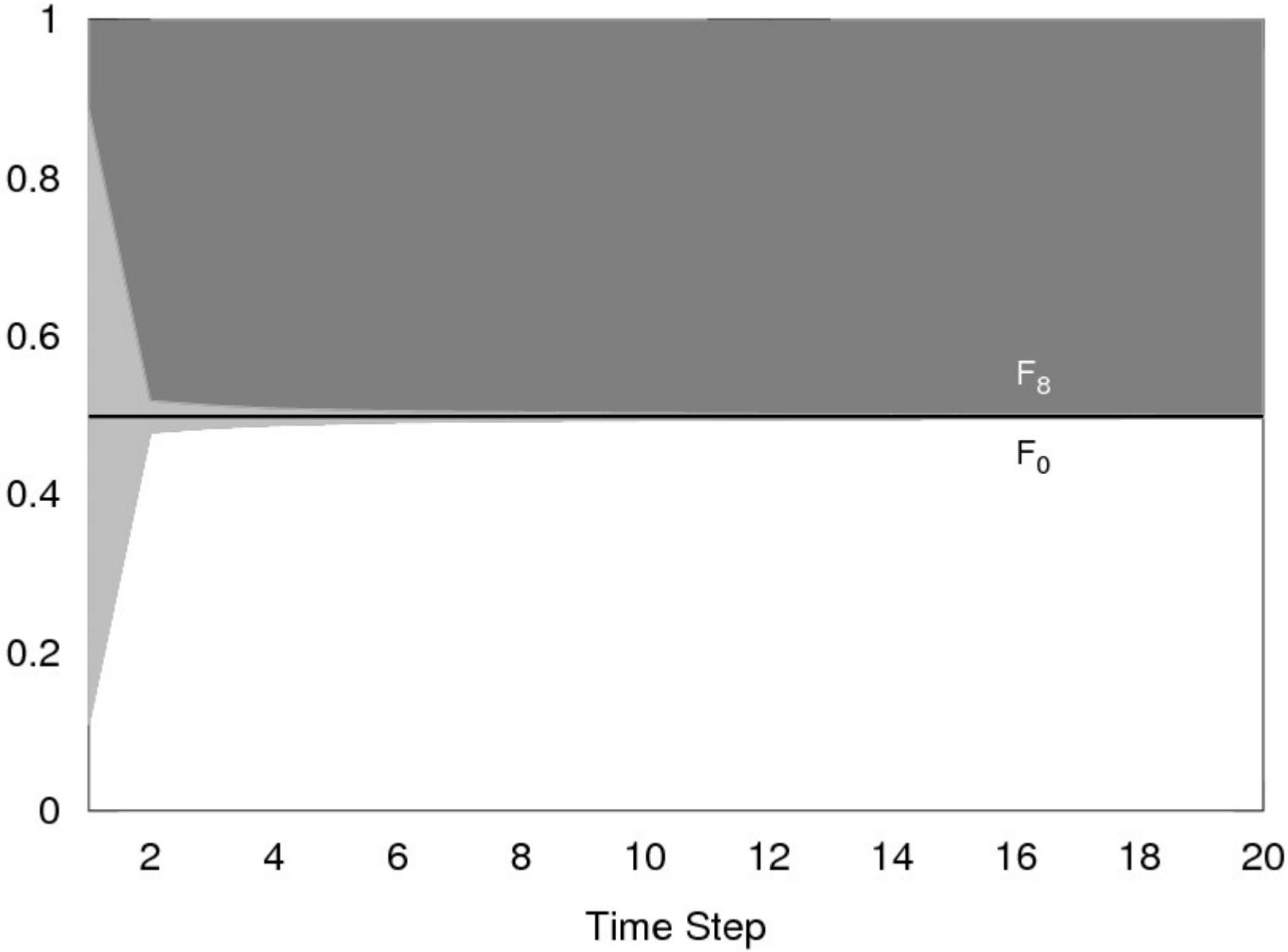
Virtual machine **MAN**ager

```
1: i ← GetProcID()

2: procedure ACTIVETHREAD
3:     loop
4:         Wait Δ
5:         for all j ∈ GetNeighbors(i) do
6:             Send ⟨H_i⟩ to j
7:             Receive ⟨H'_i⟩ from j
8:             H_i ← H'_i
9:         end for
10:    end loop
11: end procedure

12: procedure PASSIVETHREAD
13:    loop
14:        Wait for message ⟨H_j⟩ from j
15:        if (H_i > H_j) then                ▷ Pull from node j
16:            D ← min(H_j, C − H_i)
17:            Send ⟨H_j − D⟩ to j
18:            H_i ← H_i + D
19:        else                               ▷ Push to node j
20:            D ← min(H_i, C − H_j)
21:            Send ⟨H_j + D⟩ to j
22:            H_i ← H_i − D
23:        end if
24:    end loop
25: end procedure
```

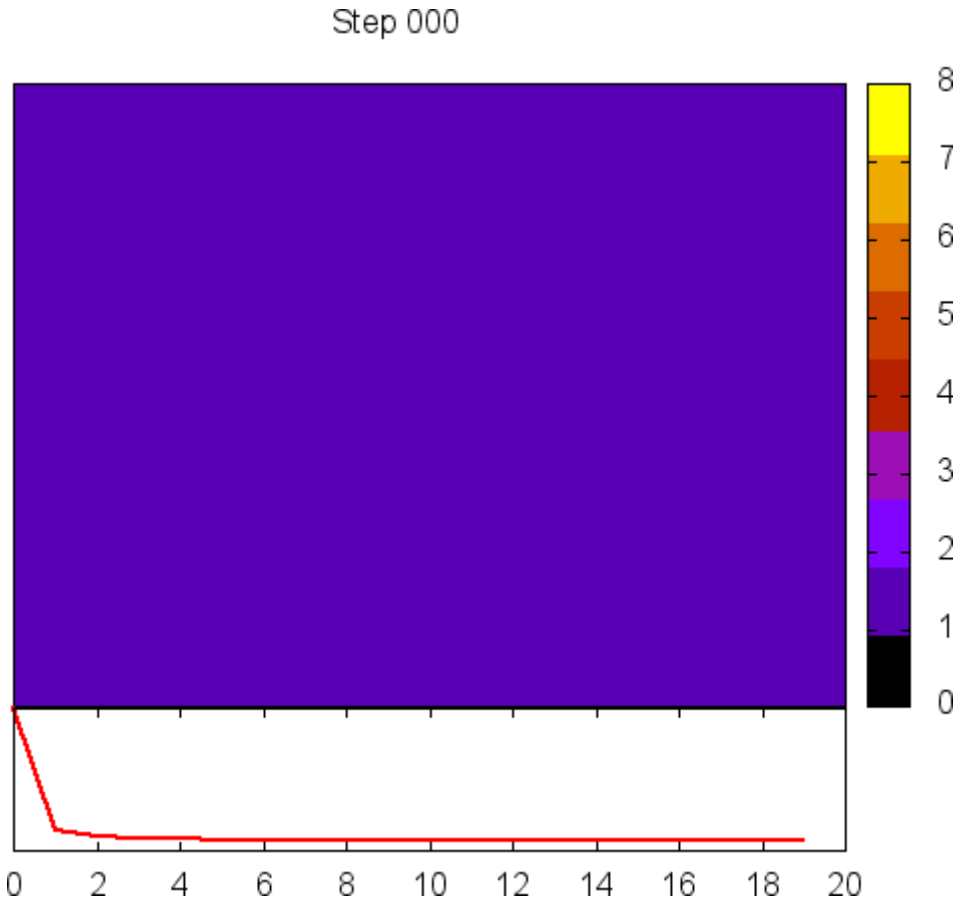
Performance assessment

- We implemented V-MAN using the cycle-driven simulator engine provided by PeerSim (peersim.sf.net)
- Parameters:
  - *K*=20 (each node maintains a list of 20 neighbors)
  - *C*=8 (maximum capacity of each host is 8 VMs)
  - Topology is managed using Newscast
  - Length of each simulation run is 20 steps
  - Results are averages of 10 independent simulation runs
- Results:
  - F_0 Fraction of empty hosts
  - $F_{0,\,opt}$ Optimal fraction of empty hosts

Experiment #1
Static System

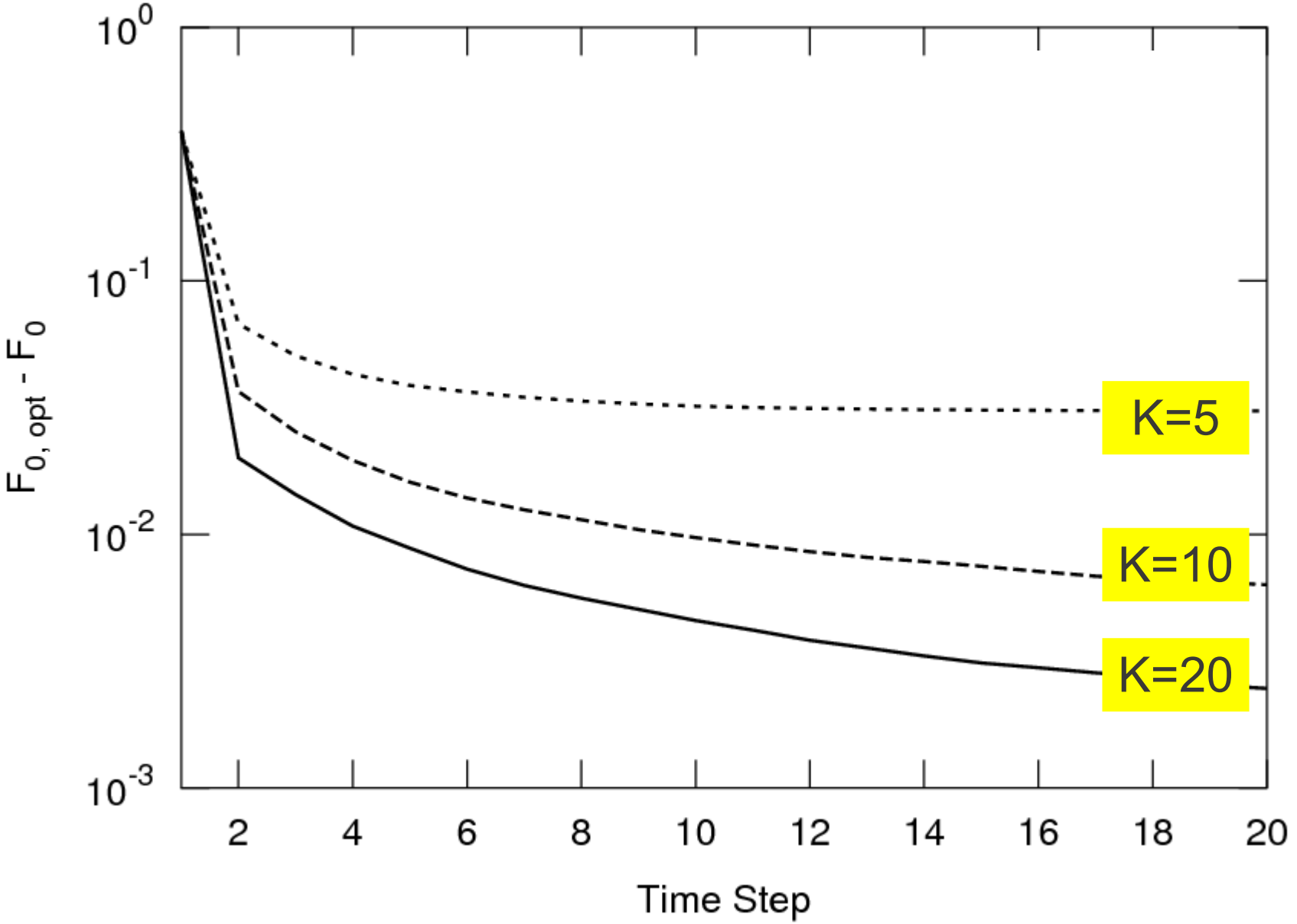


Animation

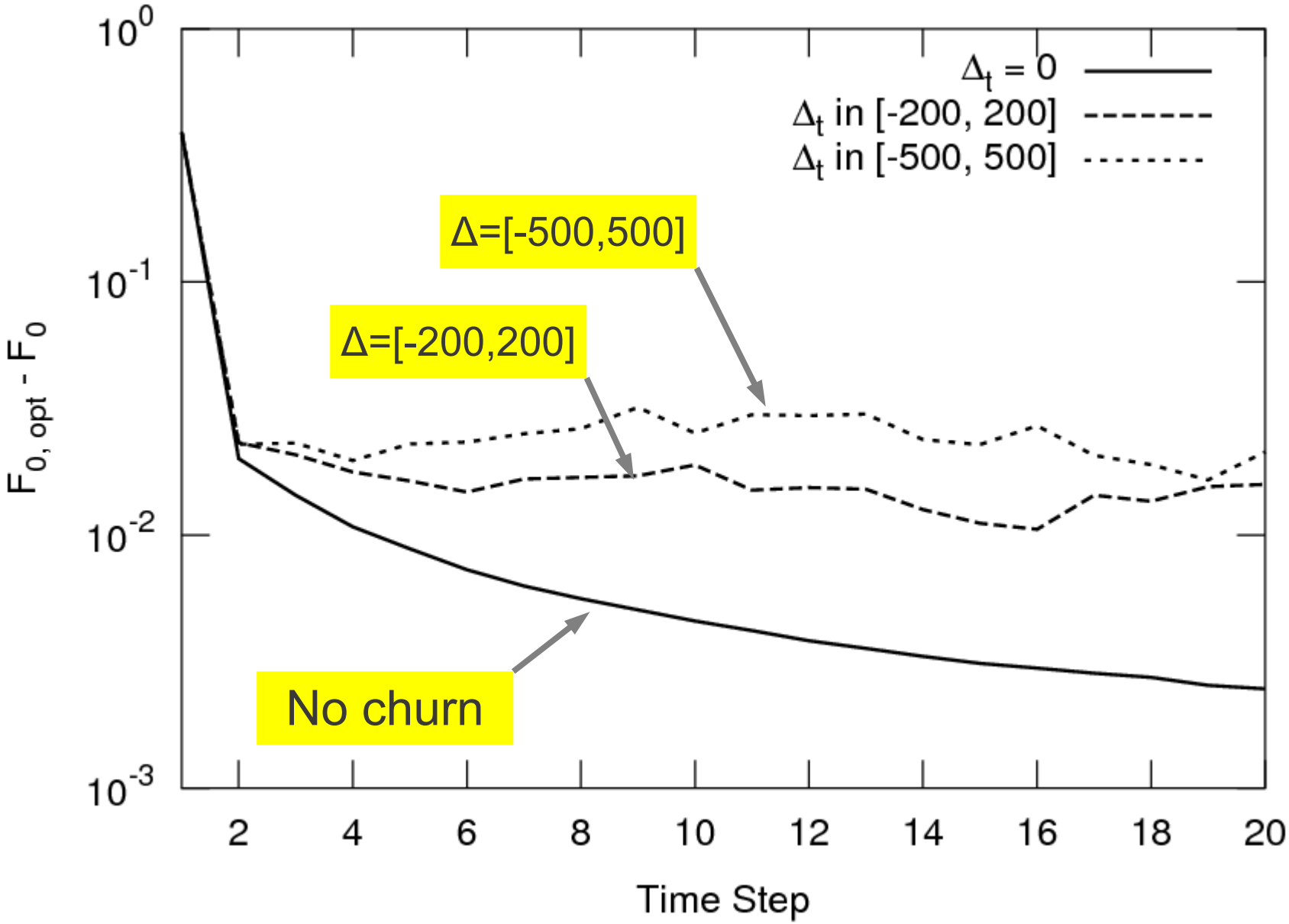


Frame-to-frame animation

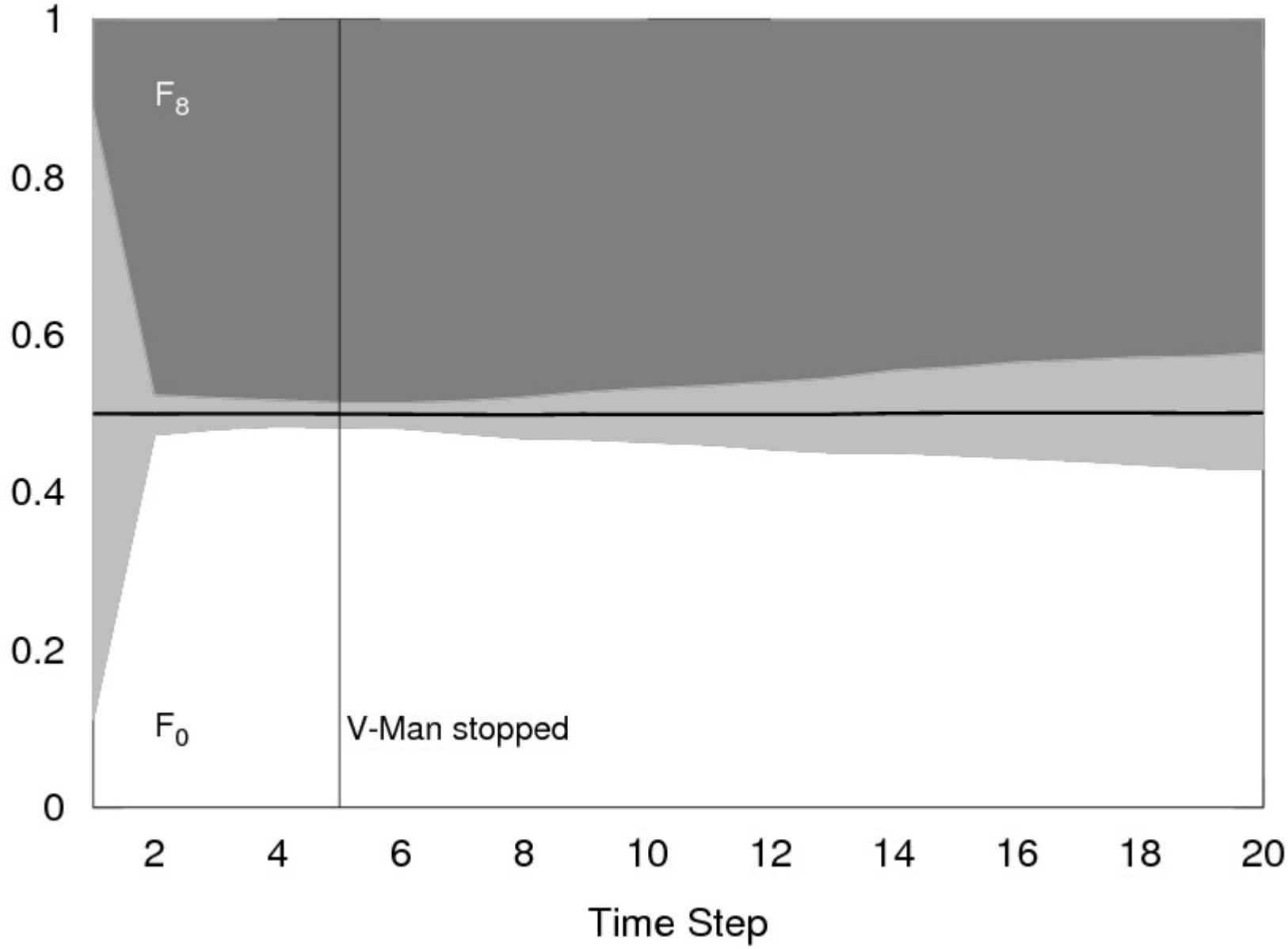
Experiment #2
Impact of View Size *K*



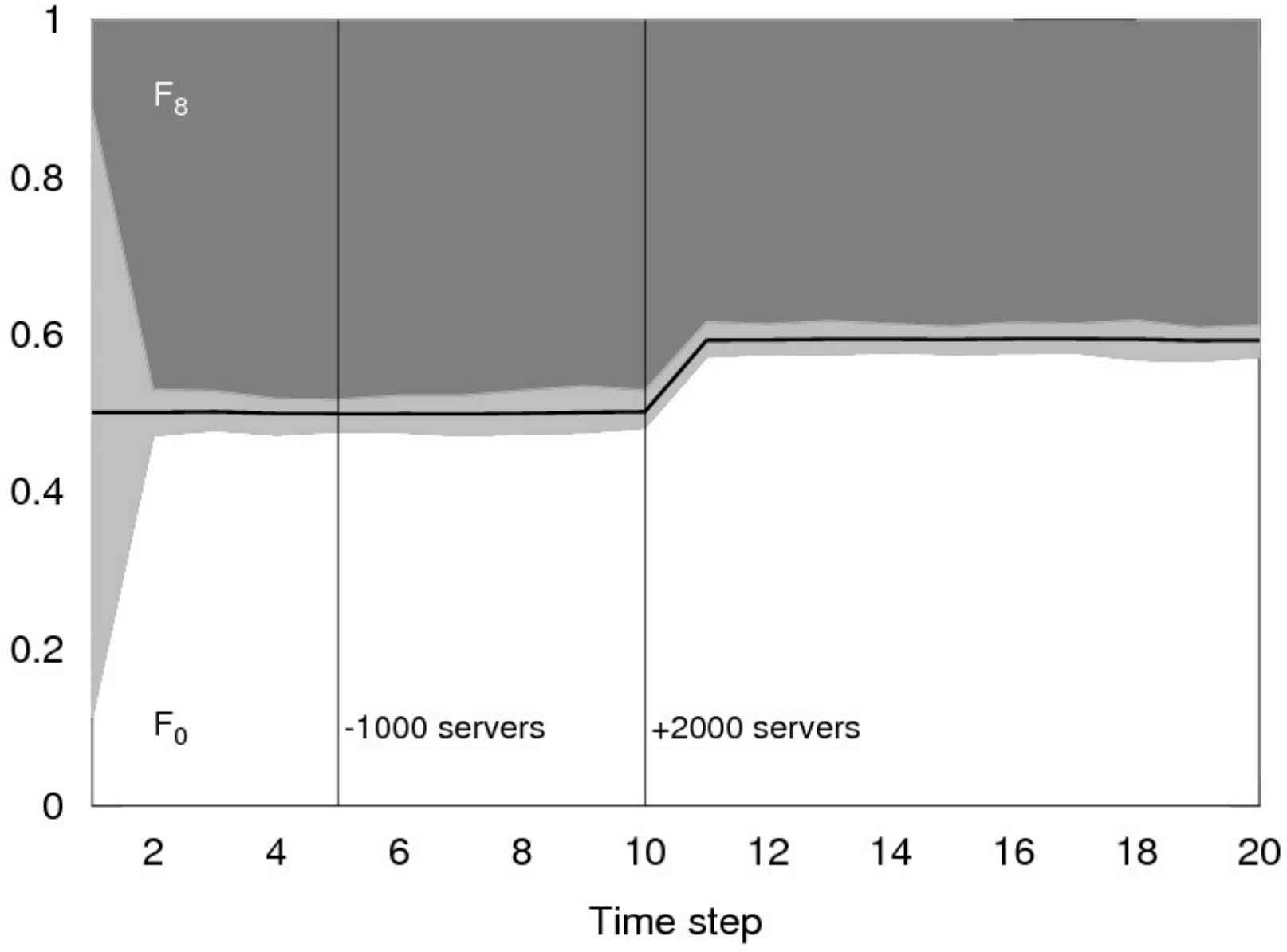
Experiment #3 / 1
Impact of Churn



Experiment #3 / 2
Stopping V-MAN

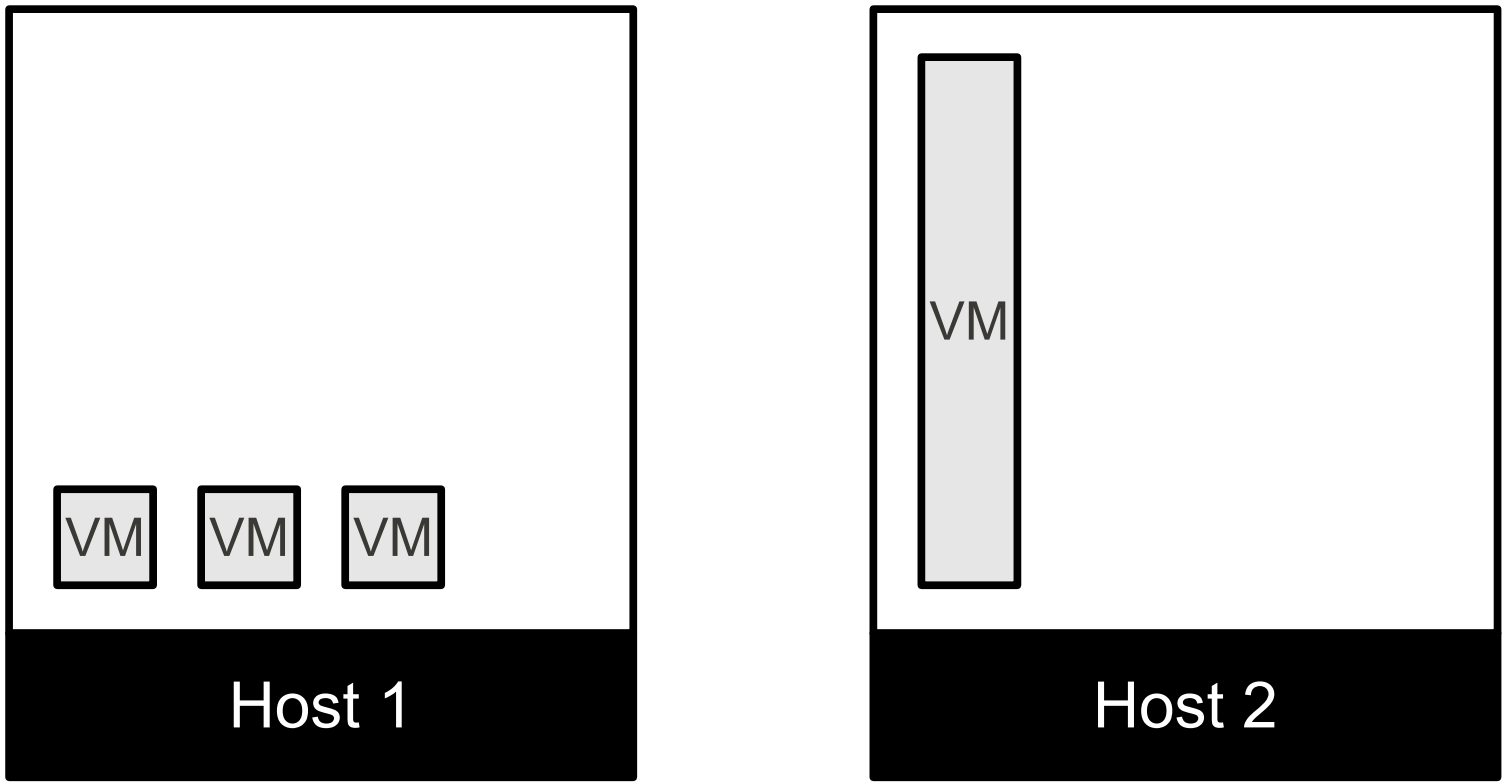


Experiment #4
Fully Dynamic System

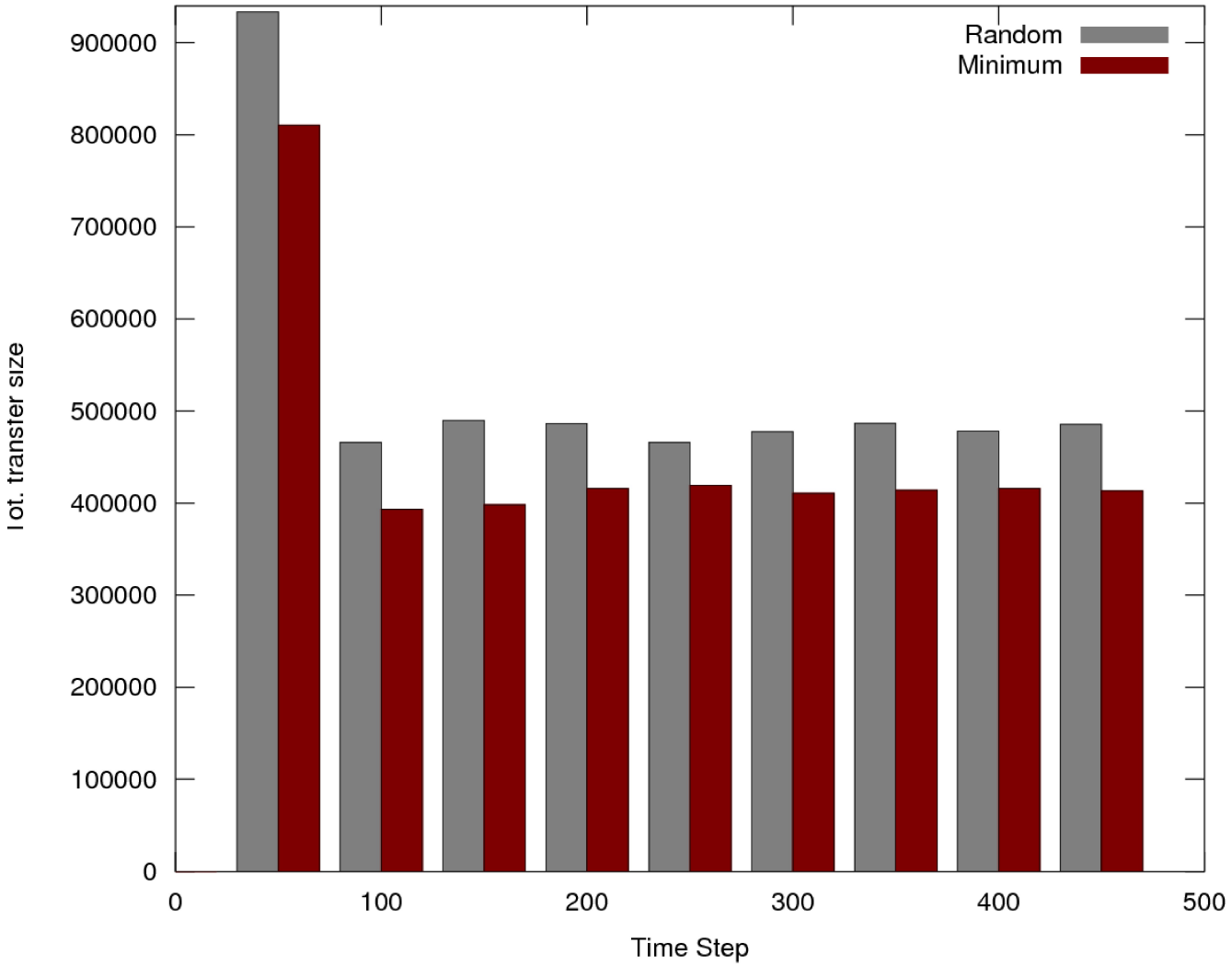


Ongoing work

- Minimize the size of transferred VM images (instead of the number of VM instances)
  - Example: what whould you migrate?



Preliminary results



Conclusions

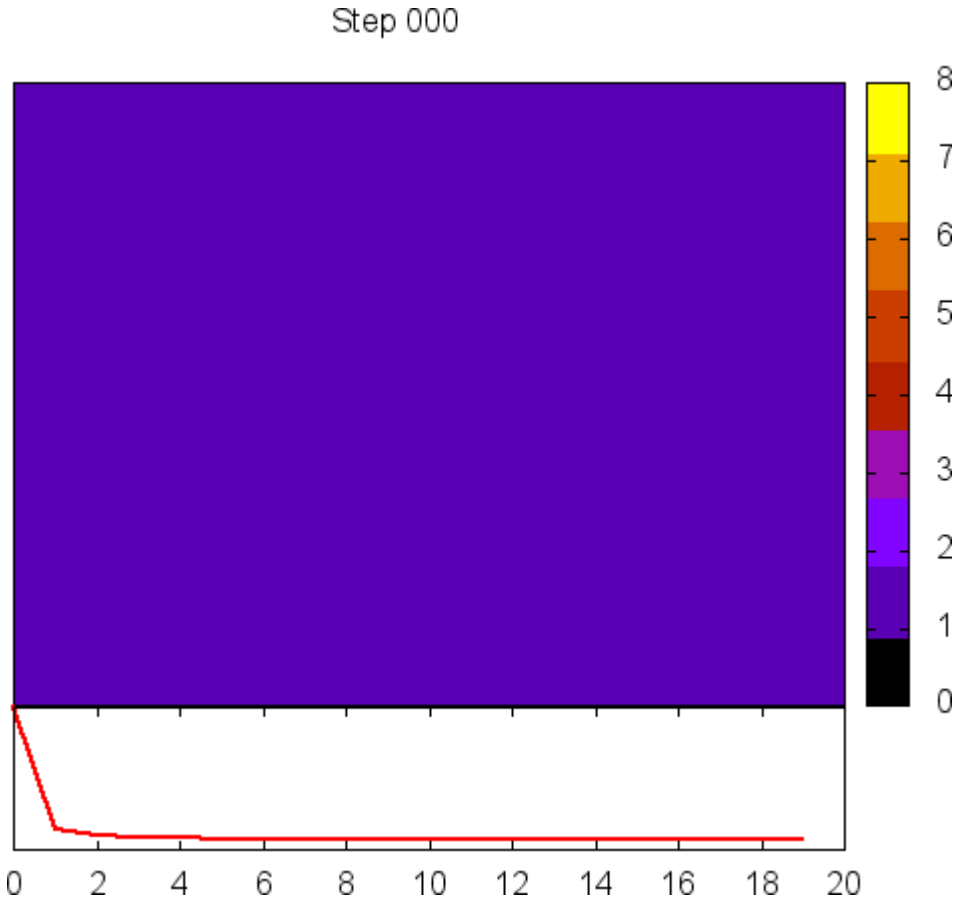
- We propose V-MAN, a totally distributed, gossip-based algorithm for VM consolidation
  - Very fast convergence speed (<5 iterations suffices)
  - Handles churn
  - Resilient to failures
- Ongoing work
  - Minimize size of migrated VM instances (*bin packing* problem?)
  - Penalize migration of long-running VMs
  - Implement V-MAN (in C)
- Source code of V-MAN simulator: http://www.moreno.marzolla.name/publications/

Thank you for your attention!

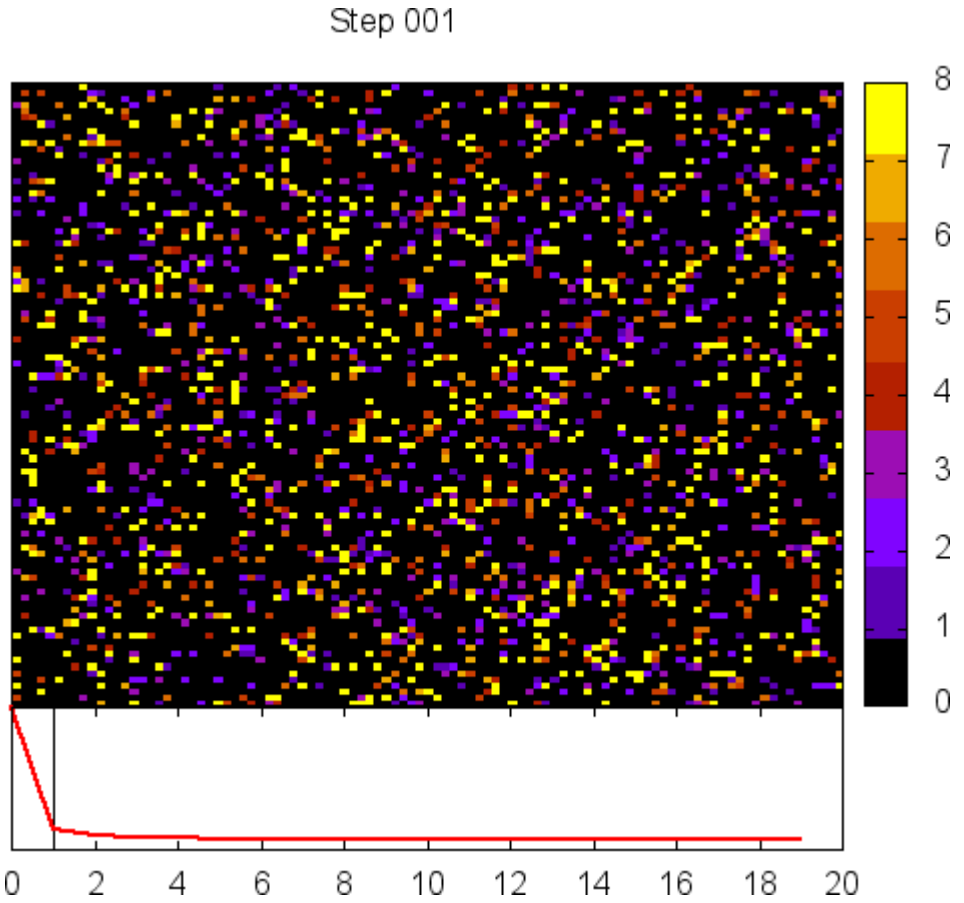
Questions?

Backup slides

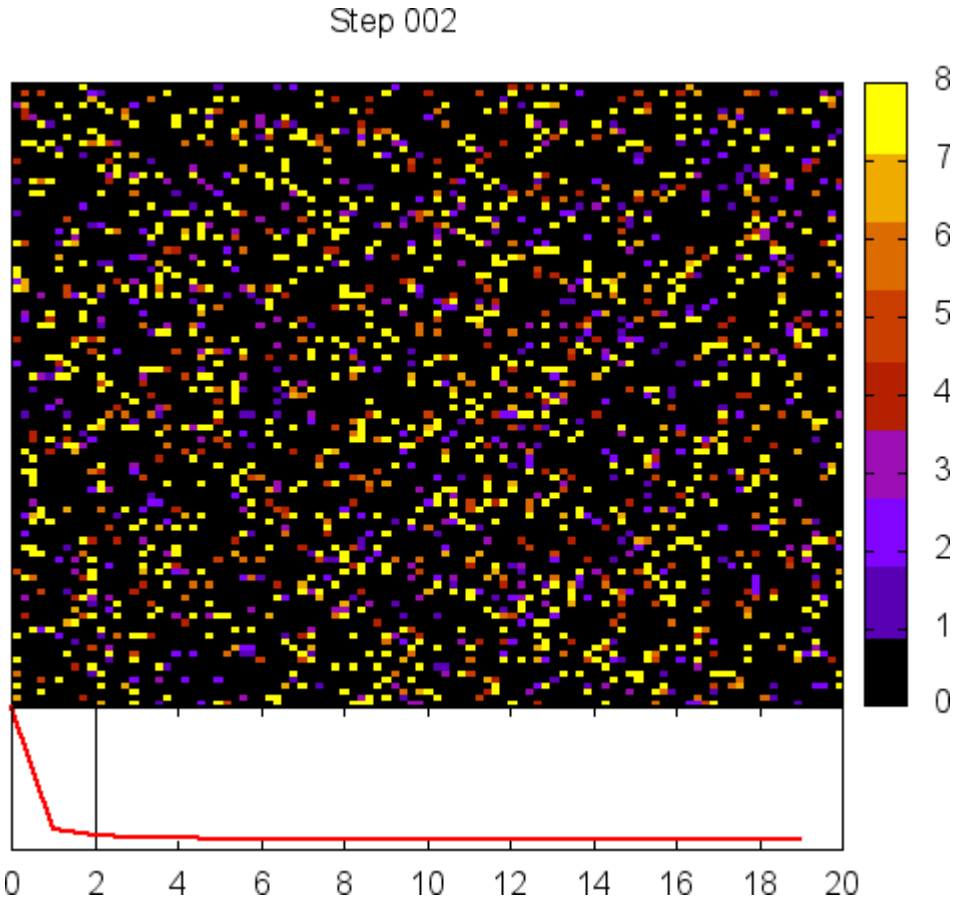
Animation



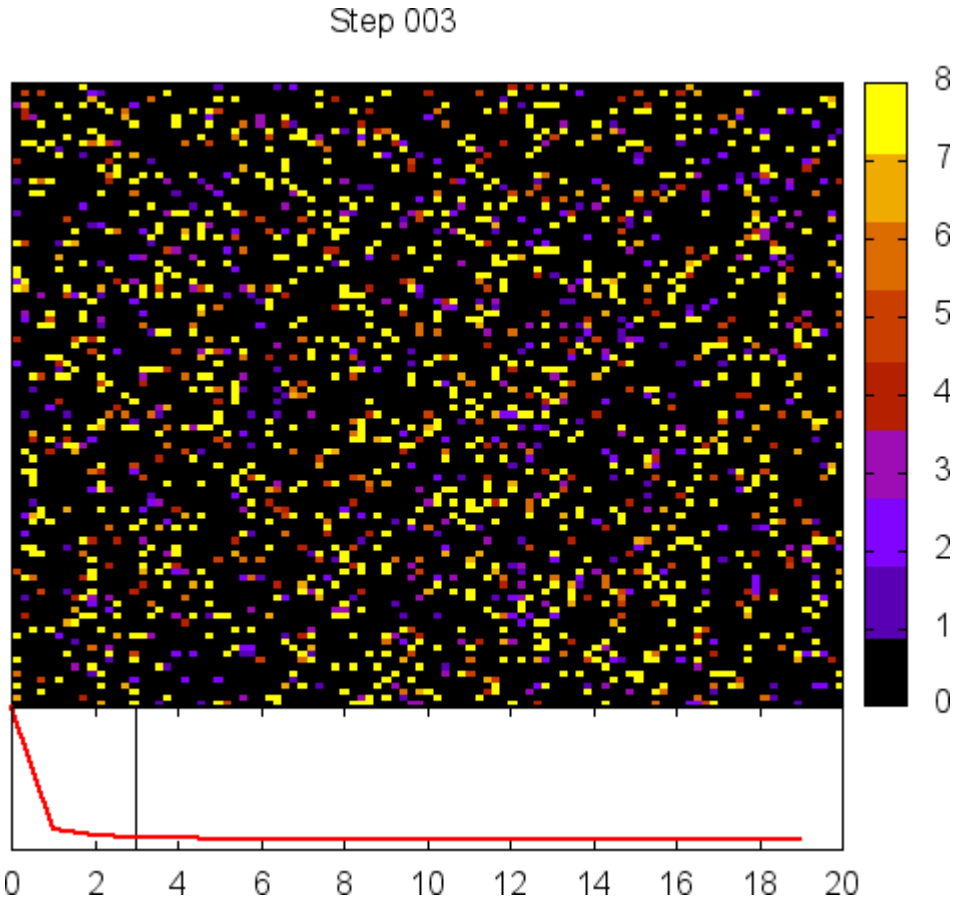
Animation



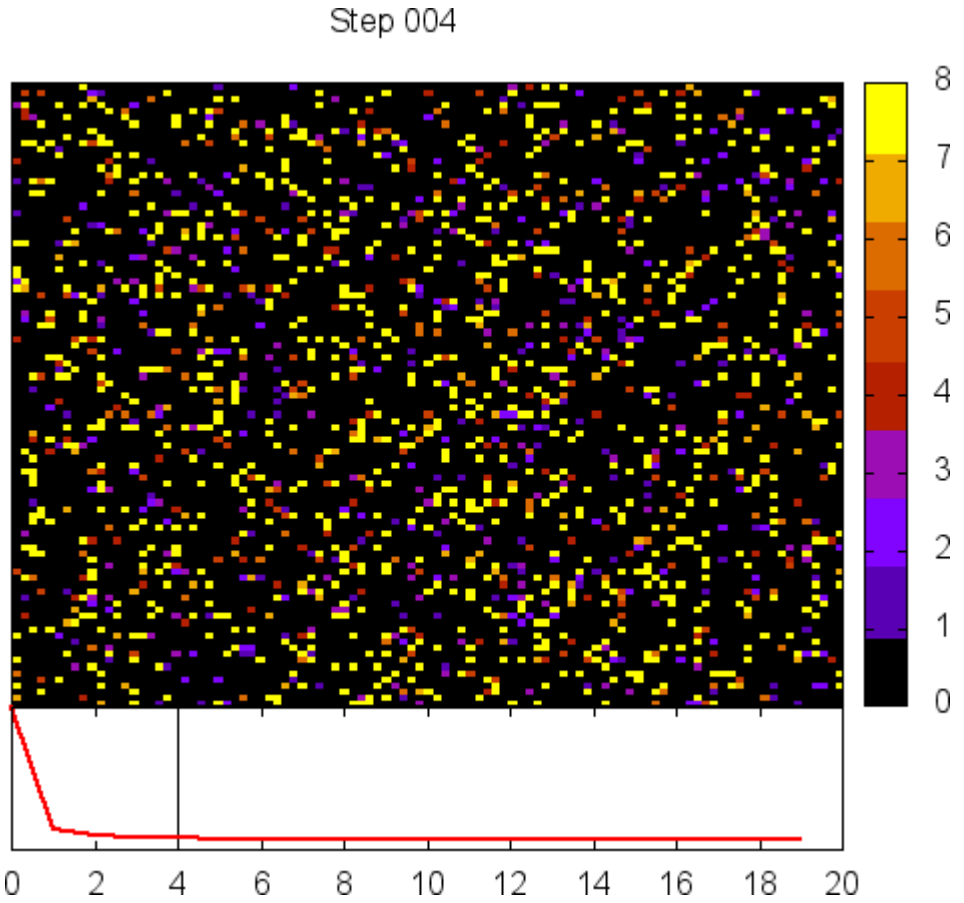
Animation



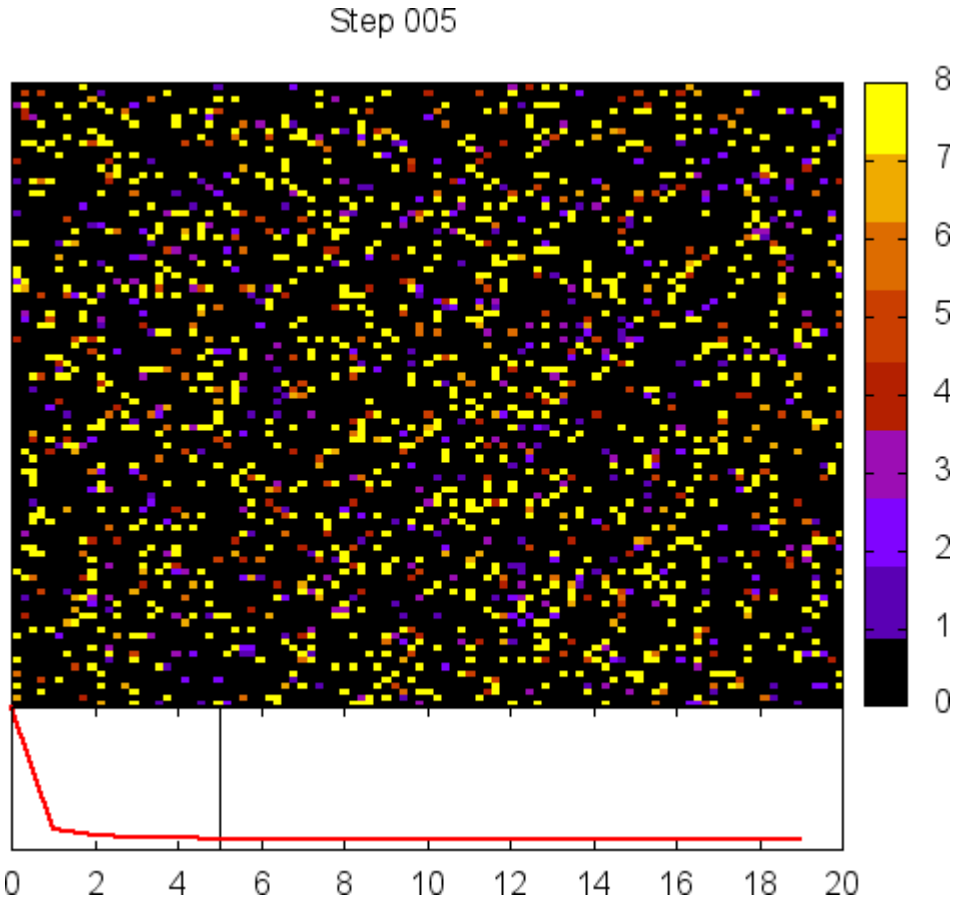
Animation



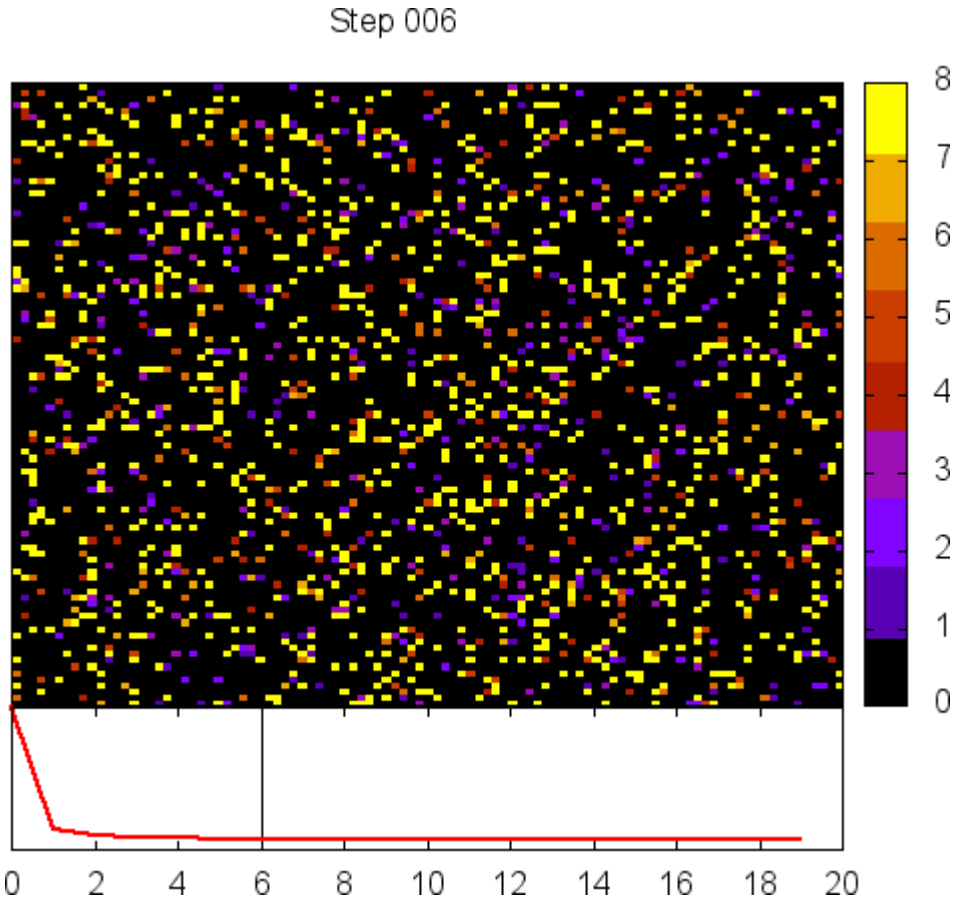
Animation



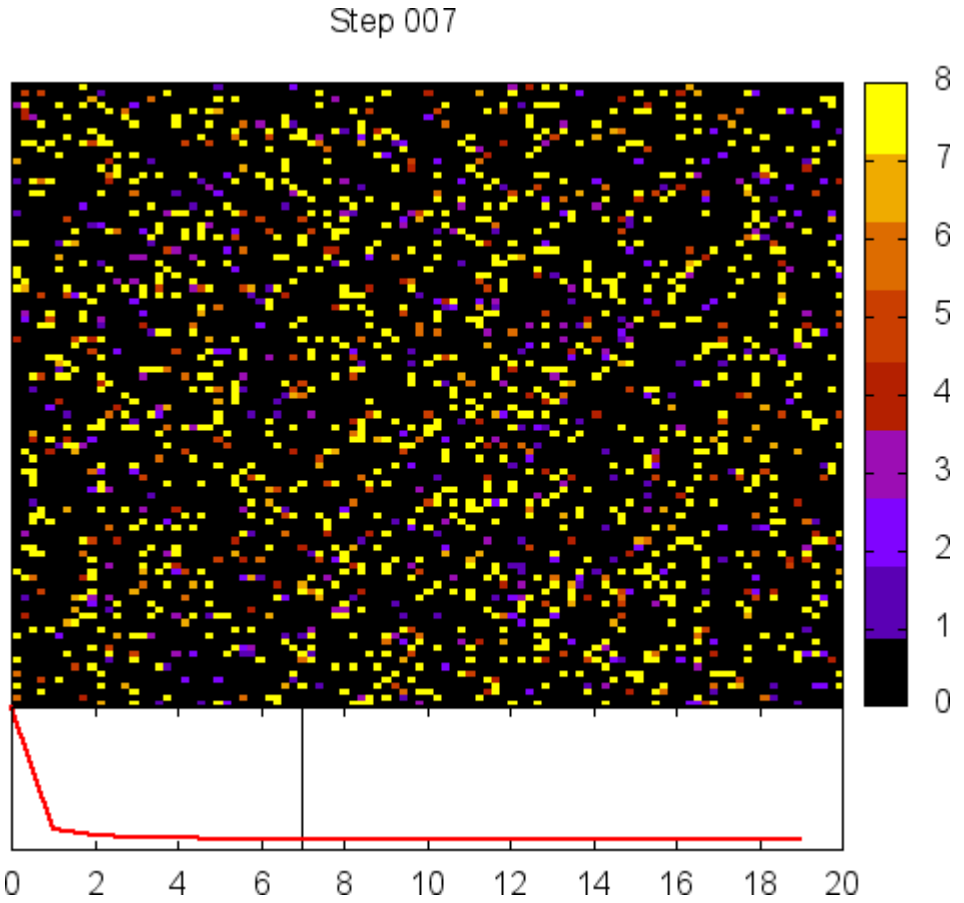
Animation



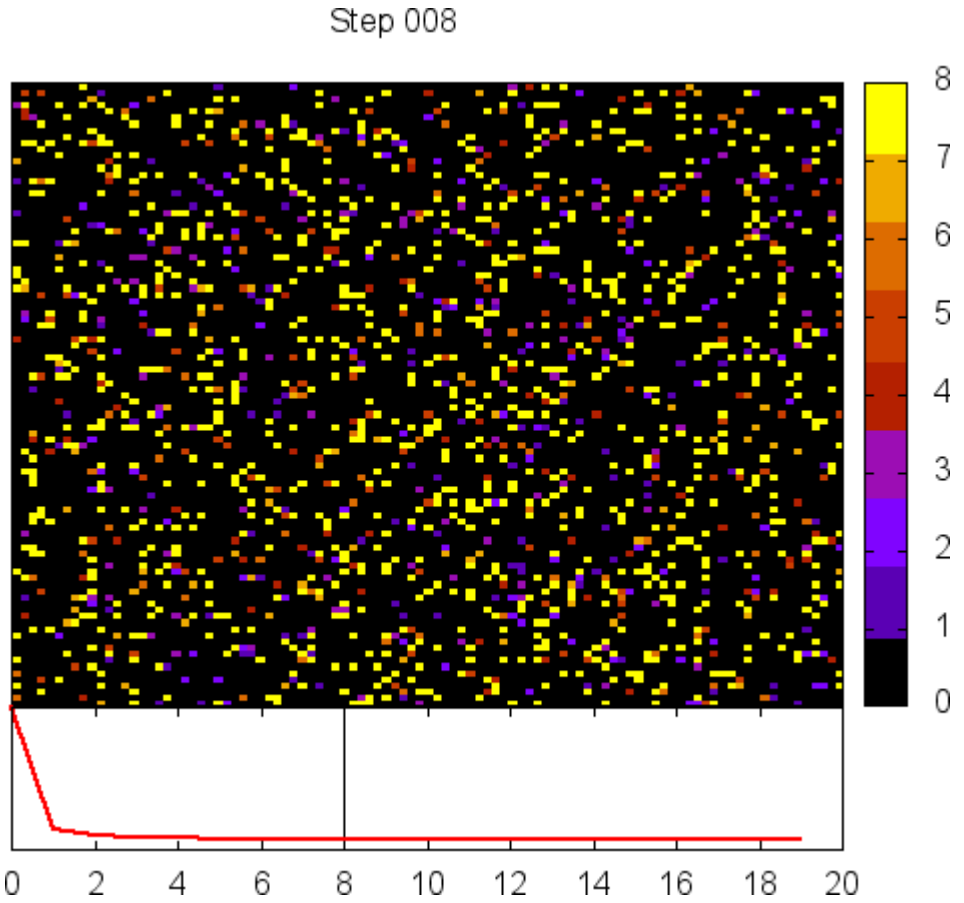
Animation



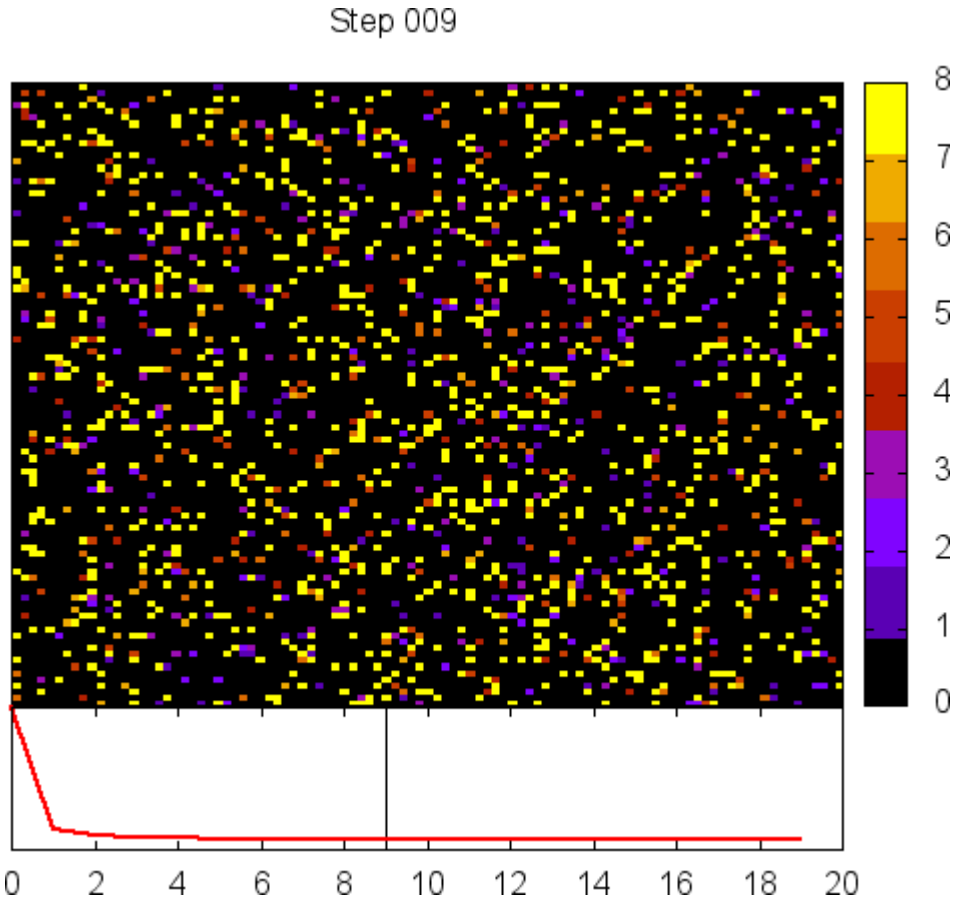
Animation



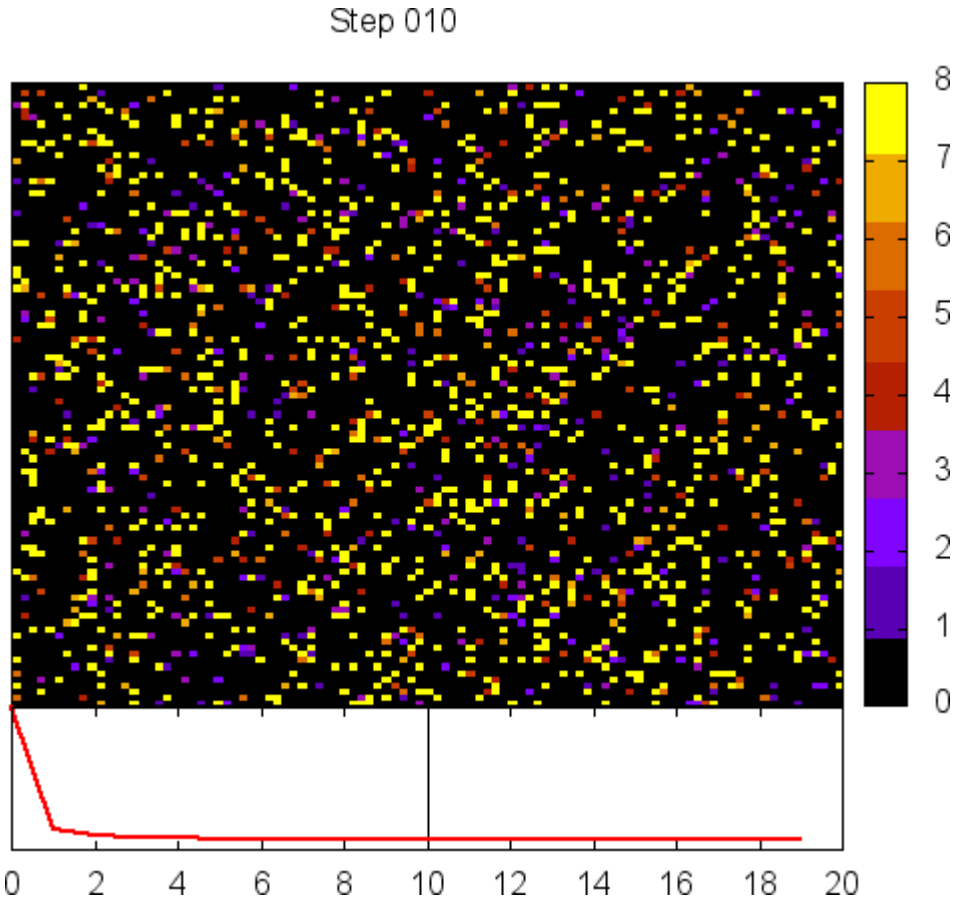
Animation



Animation



Animation



Back to presentation