Intra-Domain Pathlet Routing

Marco Chiesa **Gabriele Lospoto**
Massimo Rimondini Giuseppe Di Battista

Department of Engineering
Roma Tre University

ICCCN, 2013

Reference Scenario

- We focus on the network of an ISP (*Internet Service Provider*)
- Wide service offer (connectivity, video streaming, online office applications, etc.)
- Many requirements, for example in terms of Quality of Service and routing policies for particular traffic classes

Reference Scenario

- We focus on the network of an ISP (*Internet Service Provider*)
- Wide service offer (connectivity, video streaming, online office applications, etc.)
- Many requirements, for example in terms of Quality of Service and routing policies for particular traffic classes
- Many protocols, trying to satisfy these requirements
  - Like OSPF, RSVP-TE and MPLS
  - Each protocol offers limited, tricky or no control of routing paths

Reference Scenario

- We focus on the network of an ISP (*Internet Service Provider*)
- Wide service offer (connectivity, video streaming, online office applications, etc.)
- Many requirements, for example in terms of Quality of Service and routing policies for particular traffic classes
- Many protocols, trying to satisfy these requirements
  - Like OSPF, RSVP-TE and MPLS
  - Each protocol offers limited, tricky or no control of routing paths
- Increased network management complexity
- Nevertheless, ISPs do not have a complete control of their network

Goals of an ISP

- Fine-grained control of paths (to guarantee QoS and enforce SLAs)
- Support for routing policies
- Simplified network management and reduced configuration effort
- Fast recovery from network failure (for resiliency)
- Reduced impact of network changes

State of the Art

- Addressed issues
  - **Fault tolerance**: Pathlet routing [Godfrey et al., 2009], Path splicing [Motiwala et al., 2008], NIRA [Yang et al., 2007], BGP Add-paths [Van den Schrieck et al., 2010], Slick packets [Nguyen et al., 2011], YAMR [Ganichev et al., 2010]
  - **QoS satisfaction**: RSVP [Braden et al., 1997], HDP [El-Darieby et al., 2002]
  - **Support for routing policies**: HLP [Subramanian et al., 2005], MIRO [Xu et al., 2006]
  - **Scalability**: Landmark [Tsuchiya, 1988], ALVA [Behrens et al., 1998], Macro-routing [Dragos et al., 2004]
- Approaches
  - Multipath, source routing, hierarchical routing, mixed link state and path vector routing

State of the Art

- With very few exceptions (Path splicing [Motiwala et al., 2008], Slick packets [Nguyen et al., 2011]), most approaches are only applicable to interdomain routing
- Other approaches (NIRA [Yang et al., 2007], Landmark [Tsuchiya, 1988]) impose restrictions that are difficult to apply in practice

- An effective combination of approaches to pursue ISP's goals is still missing.
- **Pathlet routing** [Godfrey et al., 2009]: incorporates multipath and source-routing, and many of its principles can be borrowed in an intra-domain scenario

Pathlet Routing

- Two main concepts: *vnode* and *pathlet*

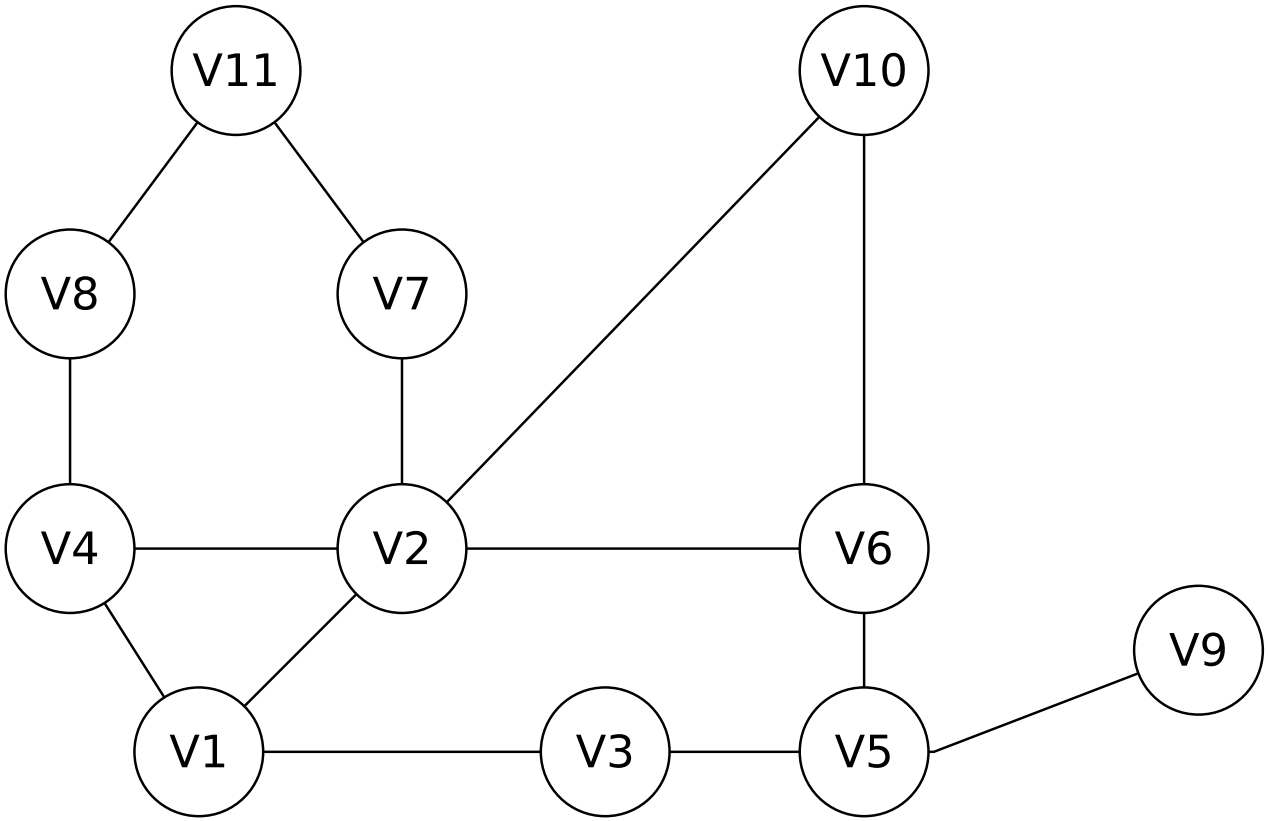
Pathlet Routing

- Two main concepts: *vnode* and *pathlet*
- **vnode**: an abstraction of a network node (can represent an AS, a group of routers, a single router, etc.)
  - In the following we assume it represents a router

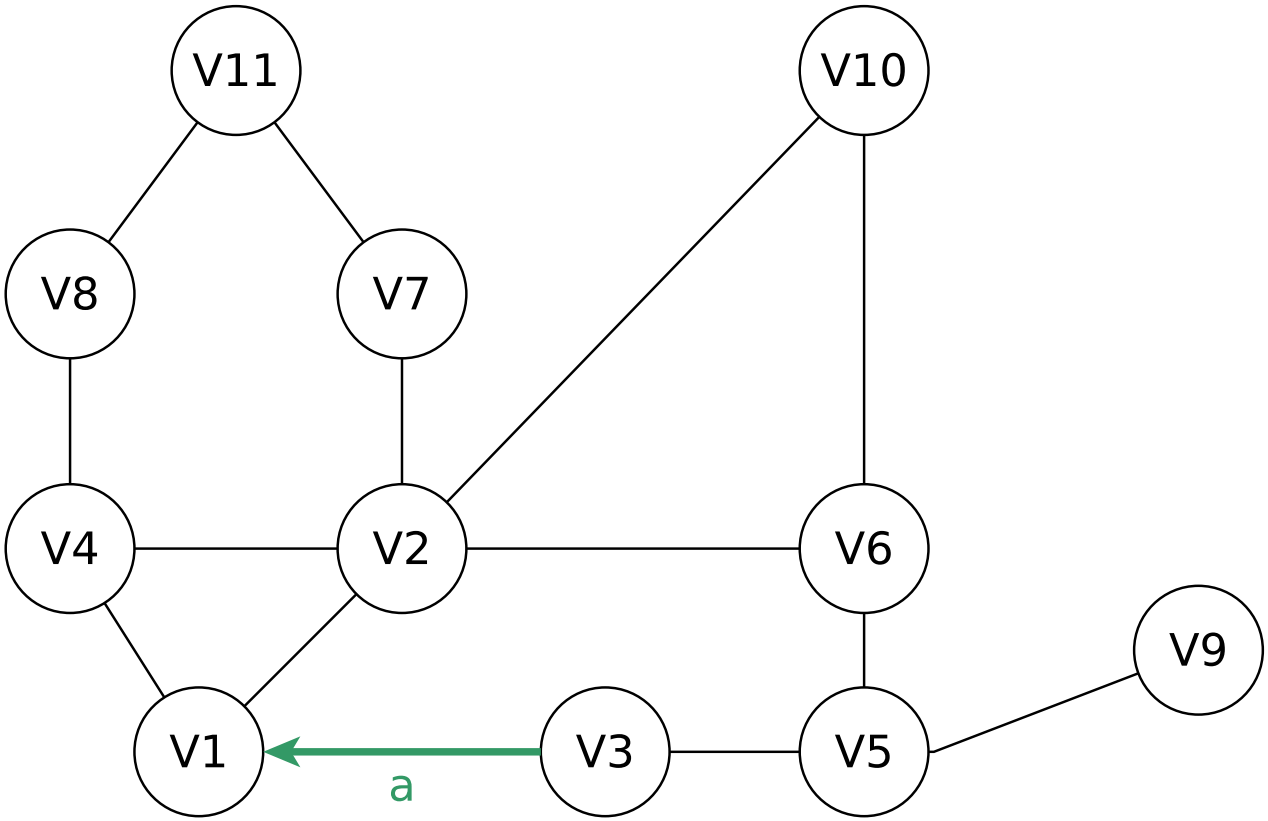
Pathlet Routing

- Two main concepts: *vnode* and *pathlet*
- **vnode**: an abstraction of a network node (can represent an AS, a group of routers, a single router, etc.)
  - In the following we assume it represents a router
- **pathlet**: a path from a vnode to another
  - It has an identifier called FID (*forward identifier*), locally unique at a vnode

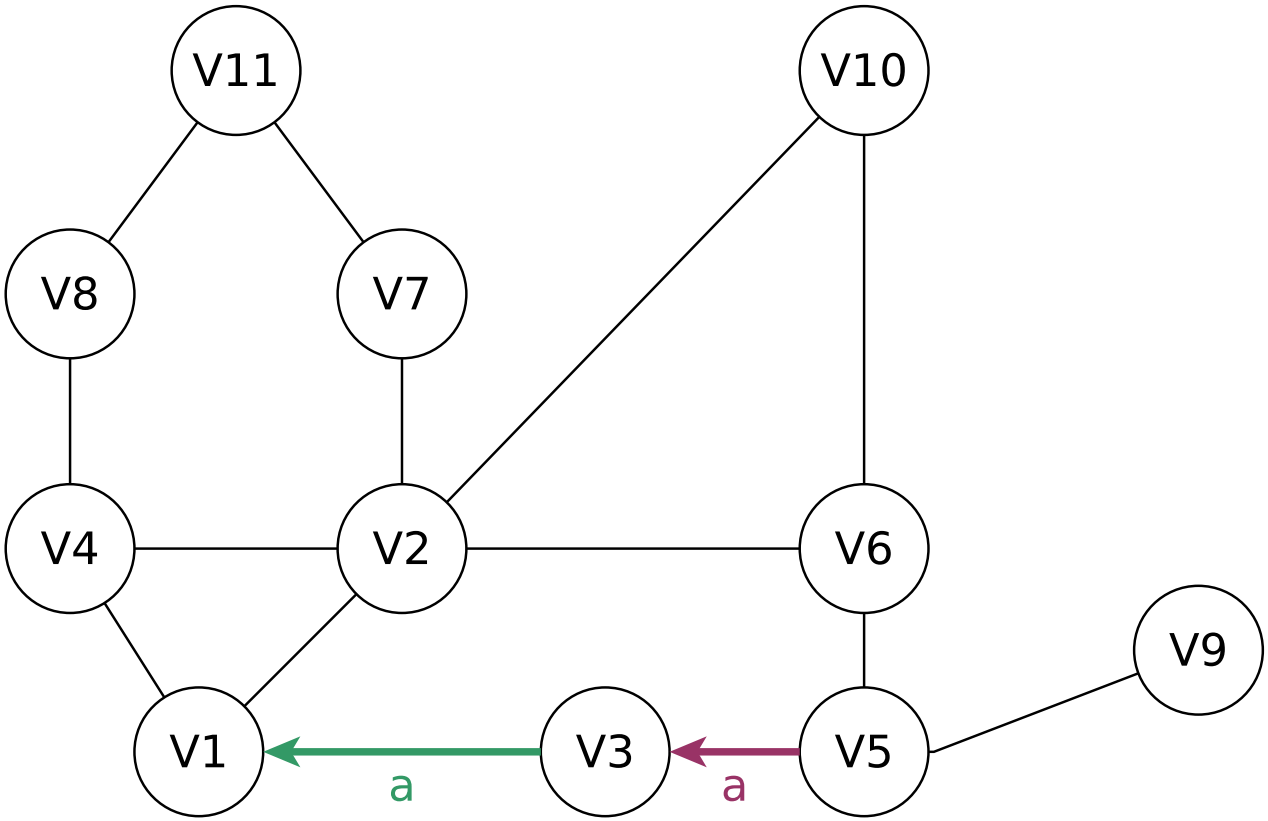
Pathlet Routing - Example



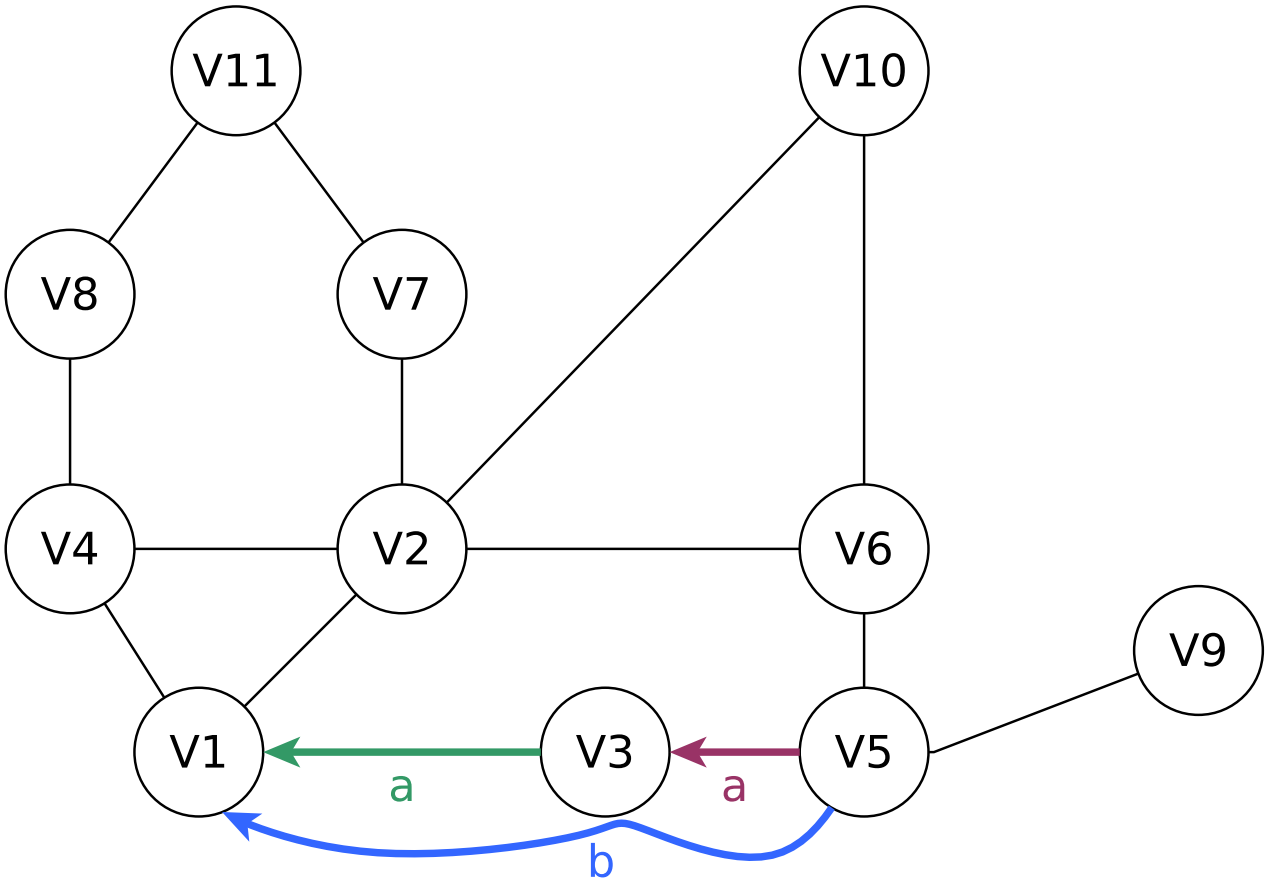
Pathlet Routing - Example



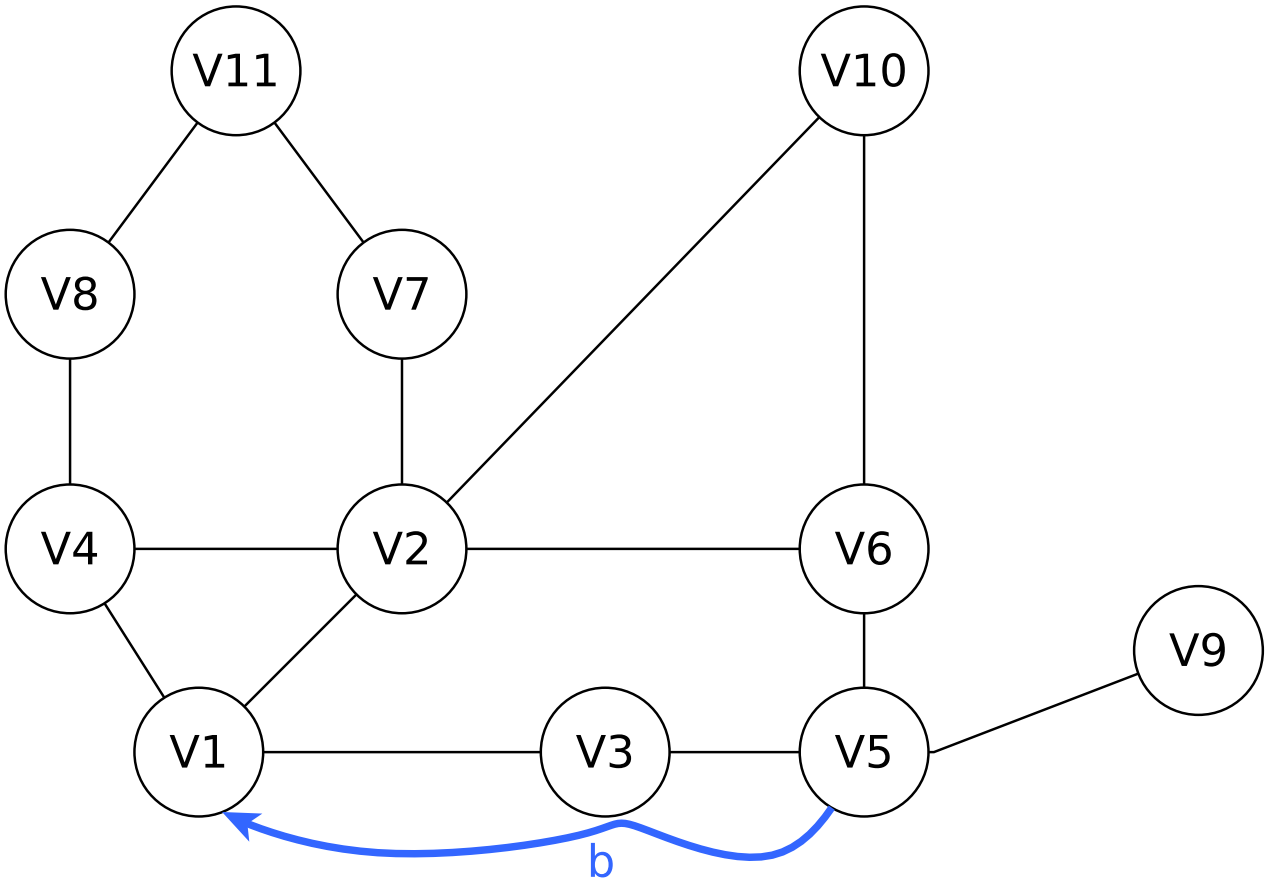
Pathlet Routing - Example



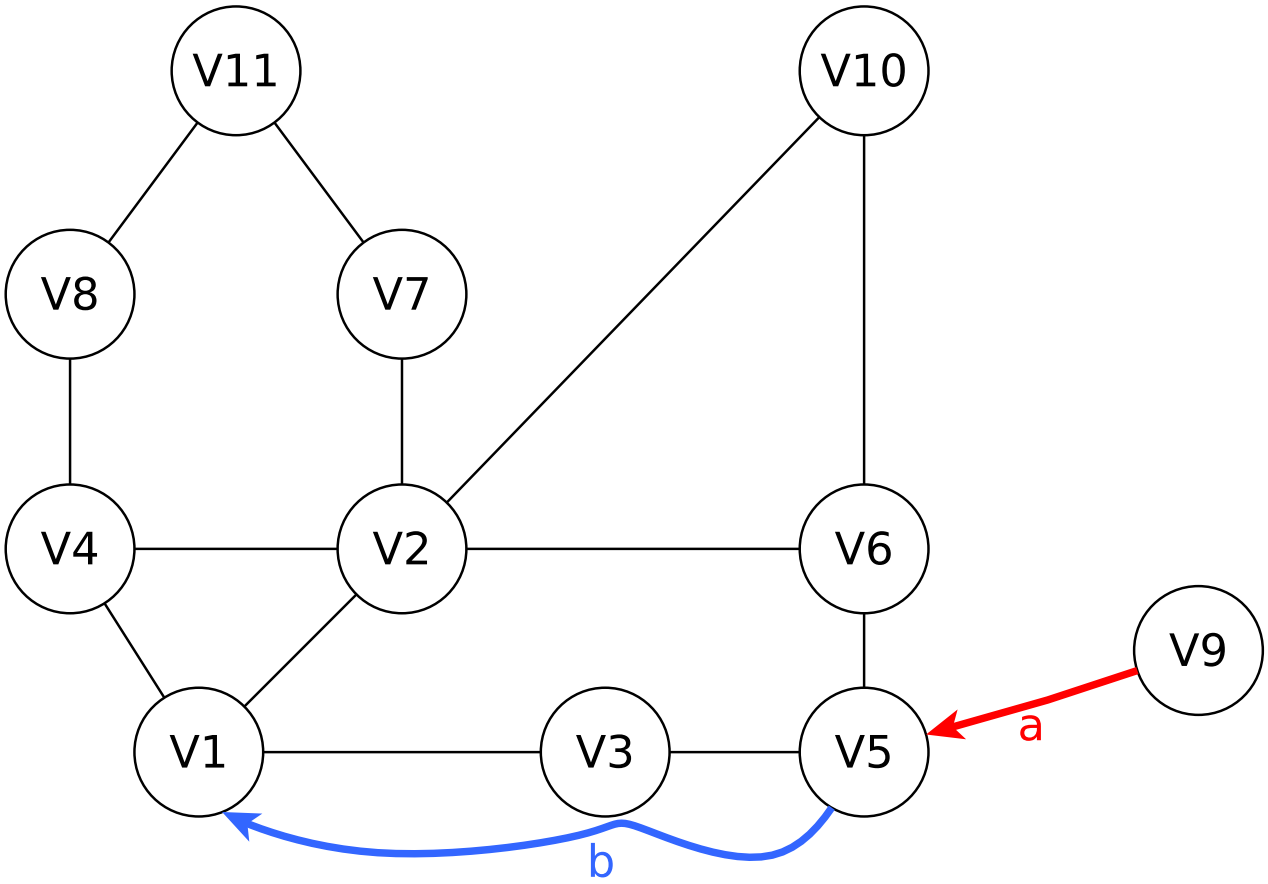
Pathlet Routing - Example



Pathlet Routing - Example



Pathlet Routing - Example



Our Contribution

- We provide a control plane for Intra-Domain Pathlet Routing
- Our control plane:
  - Provides **information hiding**, improving **scalability**
  - Improves **resiliency**
  - Allows operator to **fully control** routing paths at *different levels of granularity*

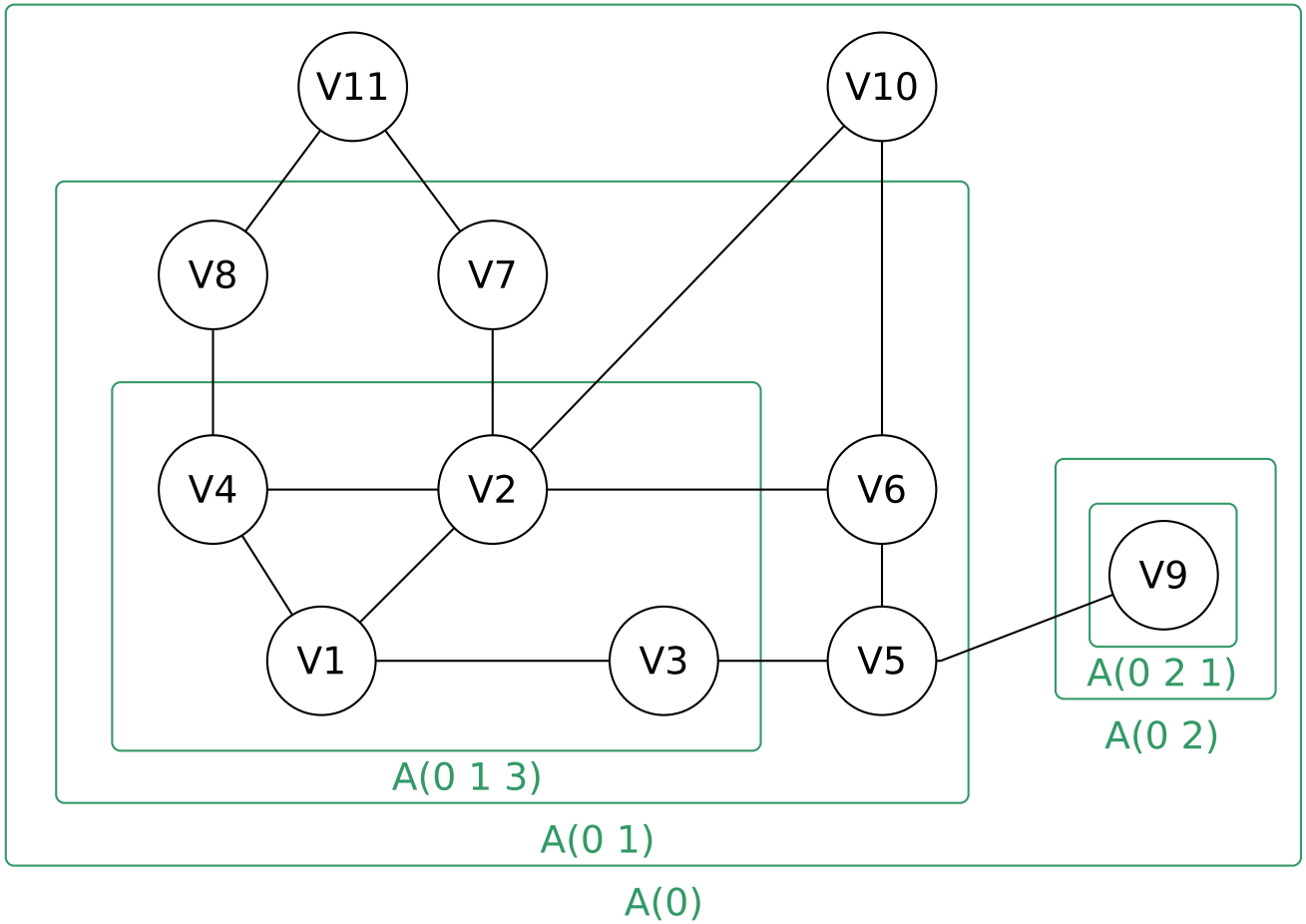
Our Contribution

- We provide a control plane for Intra-Domain Pathlet Routing
- Our control plane:
  - Provides **information hiding**, improving **scalability**
  - Improves **resiliency**
  - Allows operator to **fully control** routing paths at *different levels of granularity*
- Using, respectively:
  - **Hierarchical** network model
    - Routers are grouped into **areas** that form an arbitrary hierarchy (unlike, e.g., OSPF)
    - Propagation of routing information is constrained
  - **Multipath** routing
  - **Source routing**

Our Contribution

- We provide a control plane for Intra-Domain Pathlet Routing
- Our control plane:
  - Provides **information hiding**, improving **scalability**
  - Improves **resiliency**
  - Allows operator to **fully control** routing paths at *different levels of granularity*
- Using, respectively:
  - **Hierarchical** network model
    - Routers are grouped into **areas** that form an arbitrary hierarchy (unlike, e.g., OSPF)
    - Propagation of routing information is constrained
  - **Multipath** routing
  - **Source routing**
- Support for **Quality of Service**

An Example Network



Basic concepts

- Routing information is exchanged in the form of atoms called **Pathlet**s
- A pathlet is a t-uple $\langle FID, v_1, v_2, \sigma, \delta \rangle$

Basic concepts

- Routing information is exchanged in the form of atoms called **Pathlet**s
- A pathlet is a t-uple $\langle FID, v_1, v_2, \sigma, \delta \rangle$
  - FID: **forwarding identifier**, unique at v_1

Basic concepts

- Routing information is exchanged in the form of atoms called **Pathlet**s
- A pathlet is a t-uple $\langle FID, v_1, v_2, \sigma, \delta \rangle$
  - FID: **forwarding identifier**, unique at v_1
  - v_1: **start vertex**
  - $v_2 \neq v_1$: **end vertex**

Basic concepts

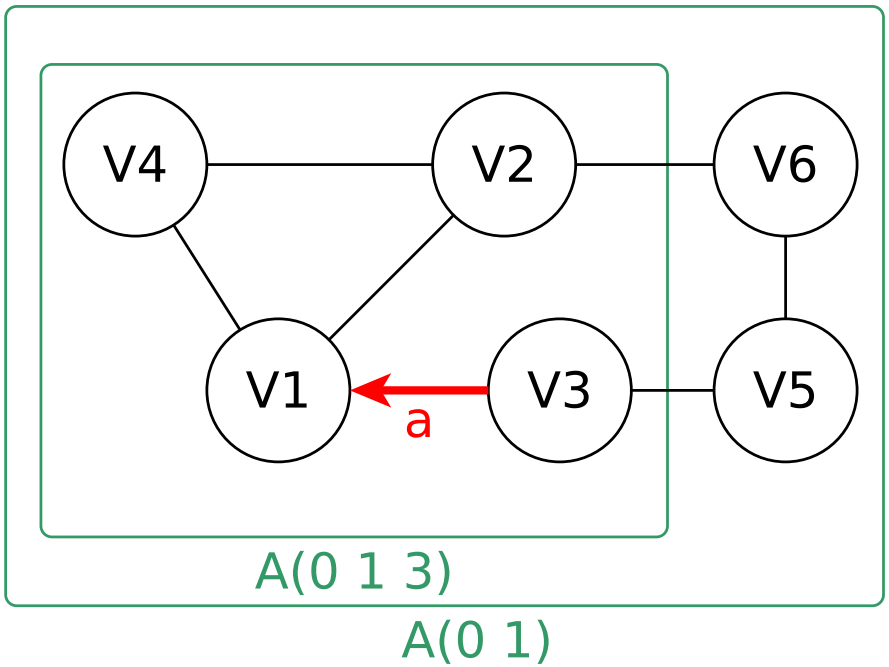
- Routing information is exchanged in the form of atoms called **Pathlet**s
- A pathlet is a t-uple $\langle FID, v_1, v_2, \sigma, \delta \rangle$
  - FID: **forwarding identifier**, unique at v_1
  - v_1: **start vertex**
  - $v_2 \neq v_1$: **end vertex**
  - σ: a stack of labels called **scope stack**
    - Influences the areas in which the pathlet is propagated

Basic concepts

- Routing information is exchanged in the form of atoms called **Pathlet**s
- A pathlet is a t-uple $\langle FID, v_1, v_2, \sigma, \delta \rangle$
  - FID: **forwarding identifier**, unique at v_1
  - v_1: **start vertex**
  - $v_2 \neq v_1$: **end vertex**
  - σ: a stack of labels called **scope stack**
    - Influences the areas in which the pathlet is propagated
  - δ: a set of network destinations available at v_2; it can be empty

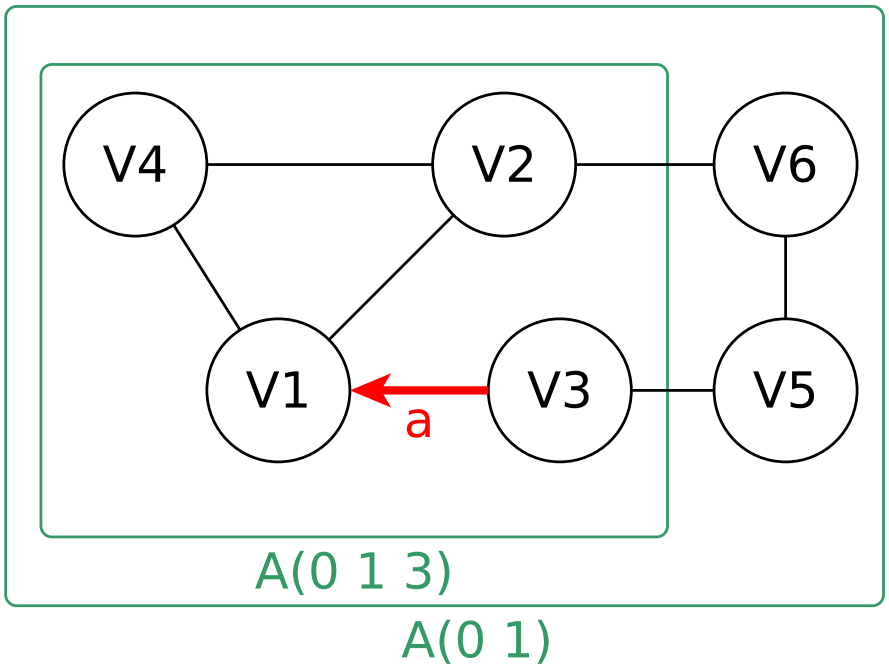
Types of Pathlet

Atomic pathlet



Types of Pathlet

Atomic pathlet

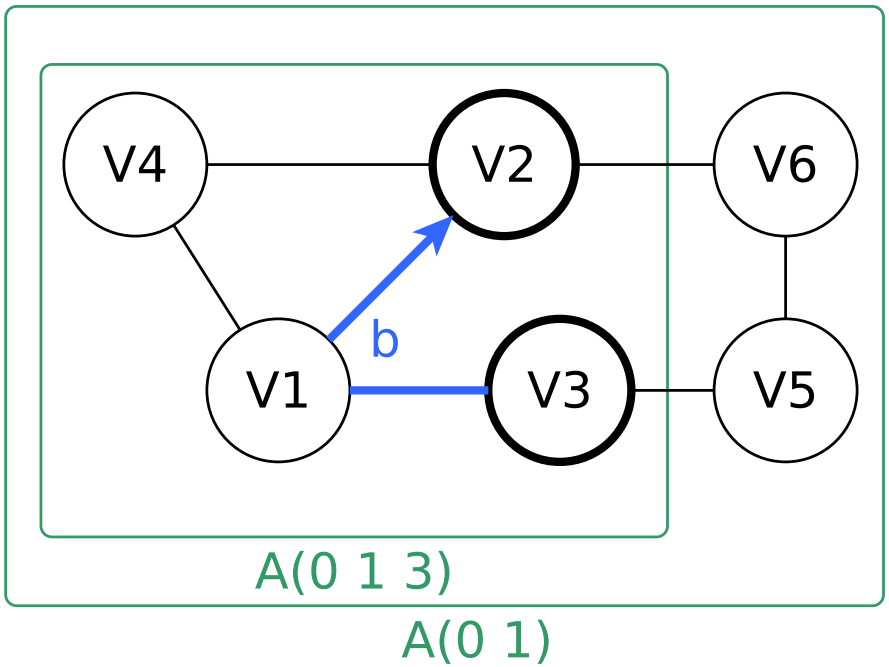


Tuple:

- *FID* : a
- Start vertex: v_3
- End vertex: v_1
- Scope stack (σ): (0 1 3 $\perp$)
- Final destinations (δ): $\emptyset$

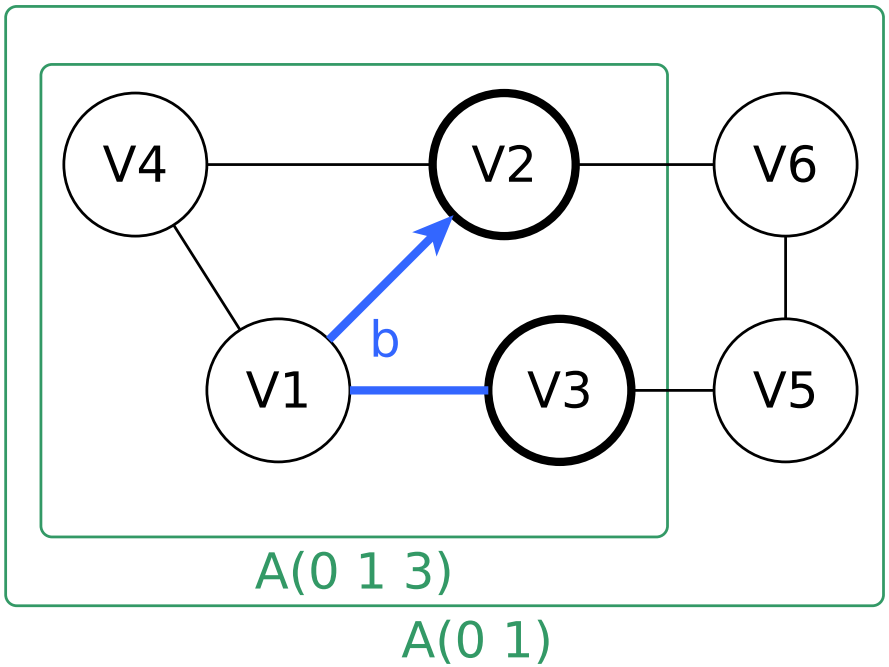
Types of Pathlet

Crossing pathlet



Types of Pathlet

Crossing pathlet

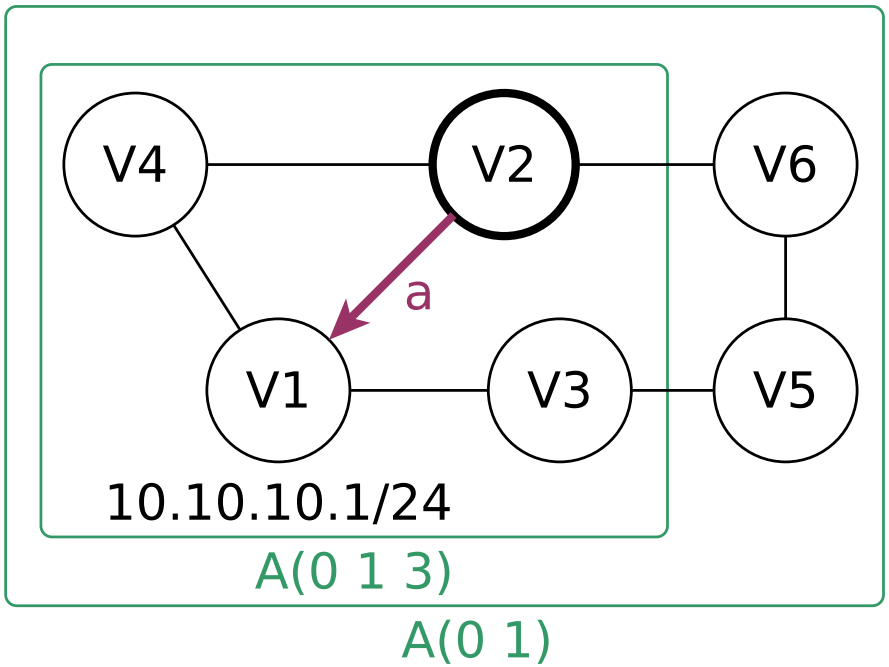


Tuple:

- *FID* : b
- Start vertex: v_3
- End vertex: v_2
- Scope stack (σ): (0 1 3)
- Final destinations (δ): $\emptyset$

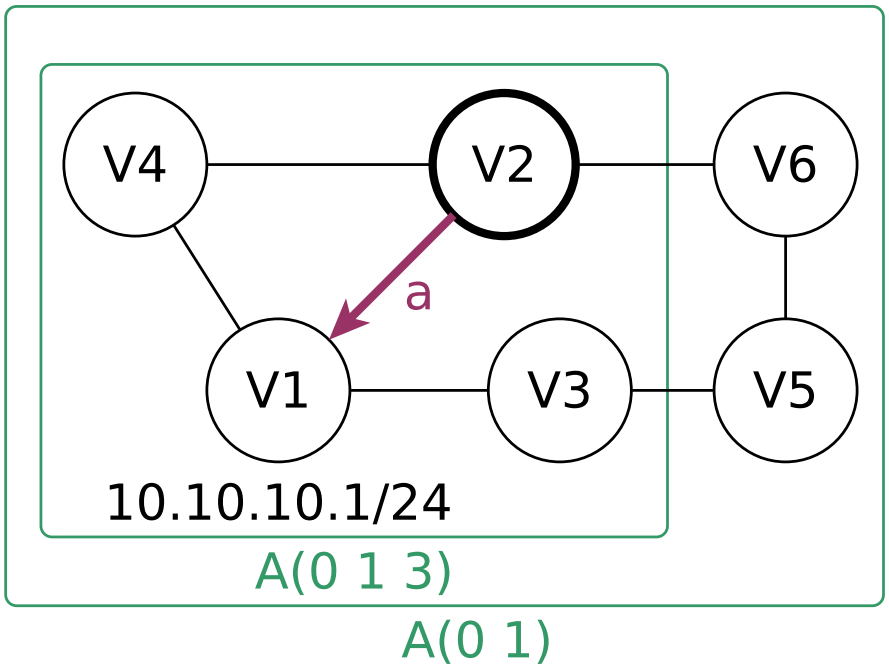
Types of Pathlet

Final pathlet



Types of Pathlet

Final pathlet



Tuple:

- *FID* : a
- Start vertex: v_2
- End vertex: v_1
- Scope stack (σ): (0 1 3)
- Final destinations (δ): {10.10.10.1/24}

Pathlet Creation and Dissemination

- Pathlet creation:
  - Each vnode creates an atomic pathlet towards each neighbor
    - Atomic pathlets represent the physical network topology

Pathlet Creation and Dissemination

- Pathlet creation:
  - Each vnode creates an atomic pathlet towards each neighbor
    - Atomic pathlets represent the physical network topology
  - Atomic pathlets are concatenated to create crossing (and final) pathlets
    - An algorithm for discovering border vertices is required to accomplish this task

Pathlet Creation and Dissemination

- Pathlet creation:
  - Each vnode creates an atomic pathlet towards each neighbor
    - Atomic pathlets represent the physical network topology
  - Atomic pathlets are concatenated to create crossing (and final) pathlets
    - An algorithm for discovering border vertices is required to accomplish this task
- Pathlet dissemination:
  - Scope stacks are assigned to the created pathlets in such a way to constrain their disseminatation
  - Each node in the network applies rules to determine where pathlets should be disseminated
    - Atomic pathlets: only within the creation area
    - Crossing (and final) pathlets: only outside the creation area

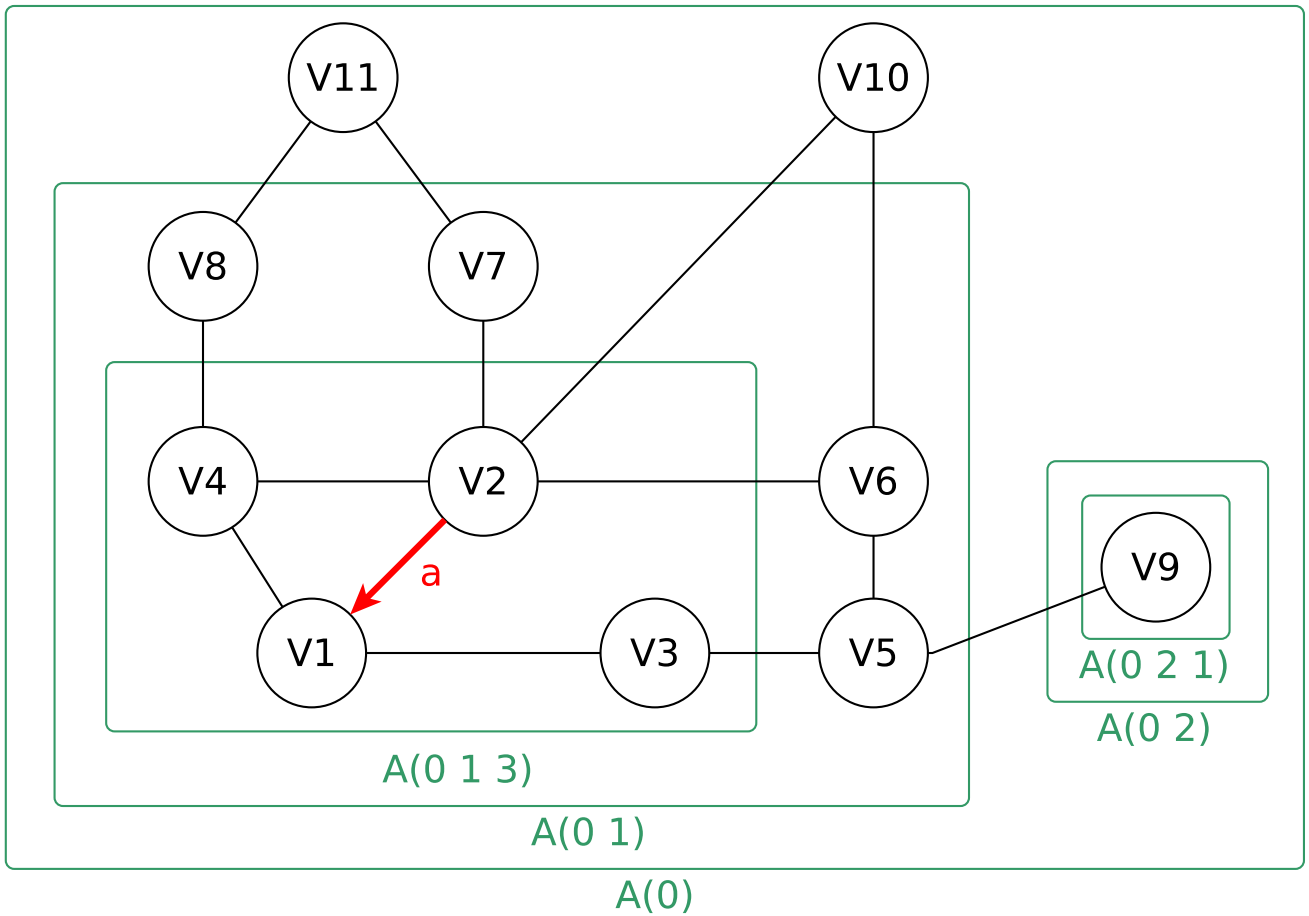
Routing Policies

- Border vertices concatenate pathlets according to **pathlet composition rules**. Examples:
  - Concatenate all the possible paths (for resilience and QoS)
  - Concatenate only one path (e.g., shortest path)
  - Preferably concatenate pathlets that do not traverse areas
- Additionally, **filters** can be defined to arbitrarily restrict the propagation of pathlets

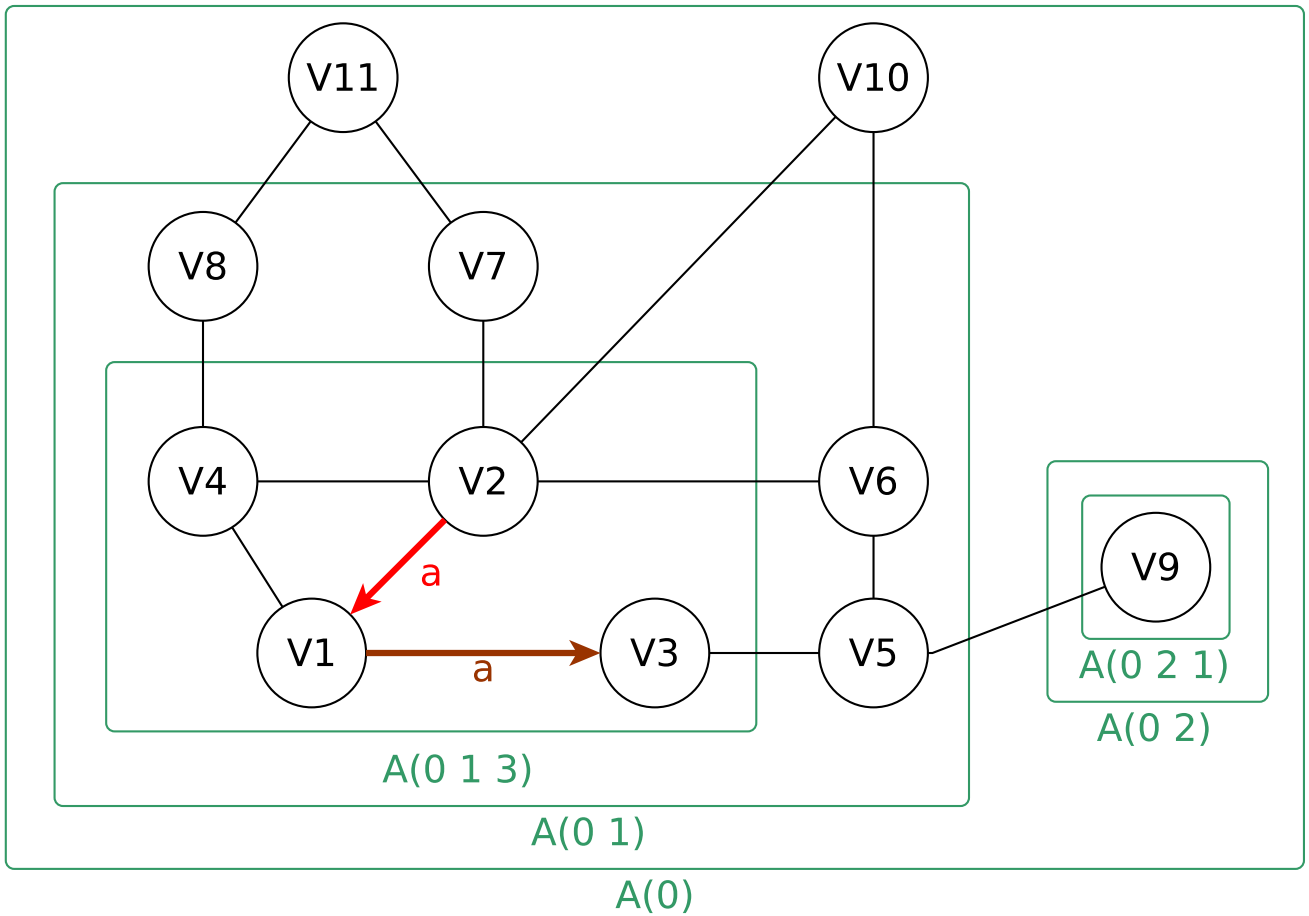
Network Dynamics

- We must handle topological changes and administrative reconfigurations
- Two main types of messages:
  - *Neighbor greetings*
    - Used to let each vertex learn about the presence and configuration of its neighbors
    - Support detection of topological and configuration changes.
  - *Pathlet dissemination*
    - Used to propagate pathlets
    - Used to withdraw no longer available pathlets

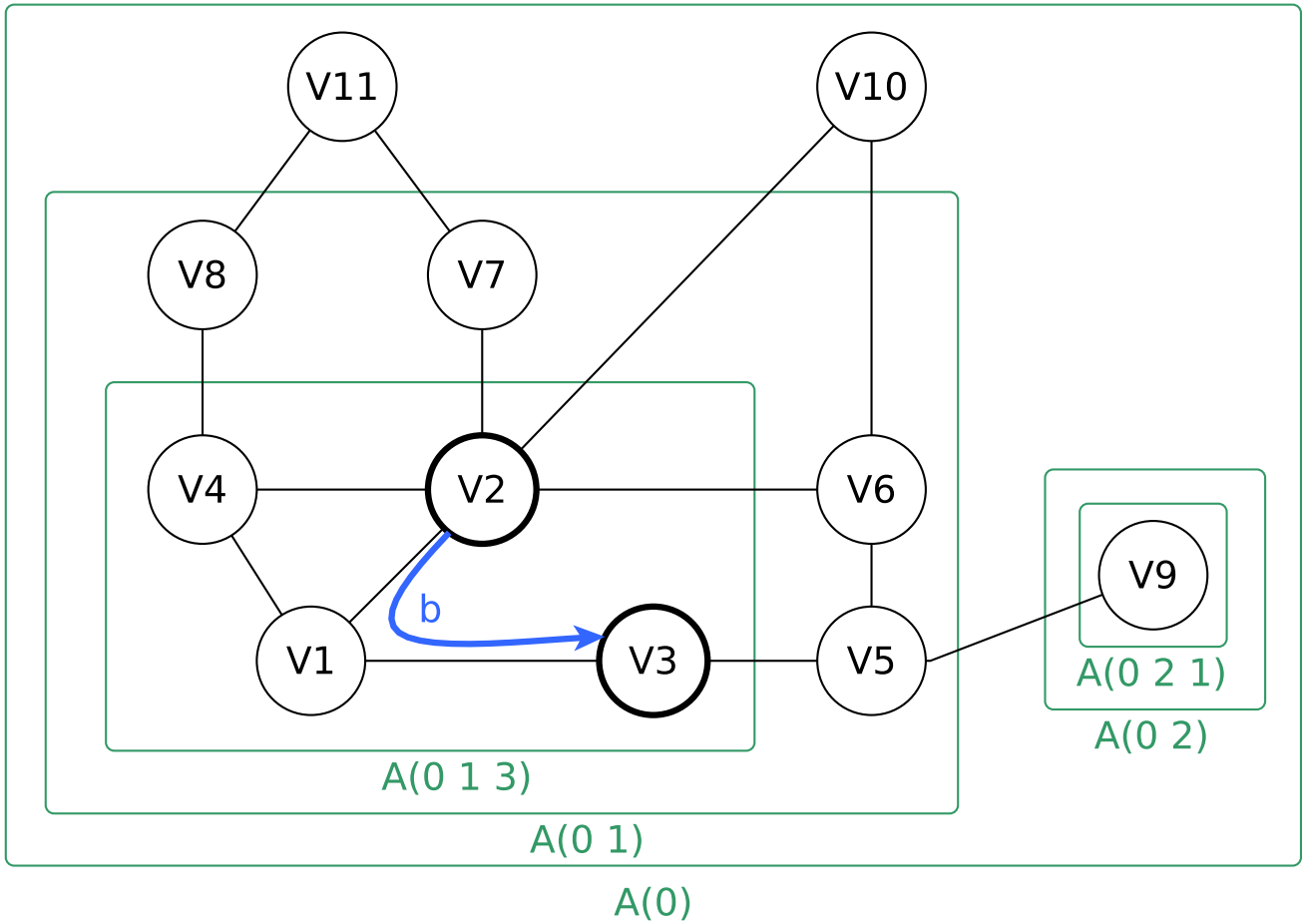
Network Dynamics: Example



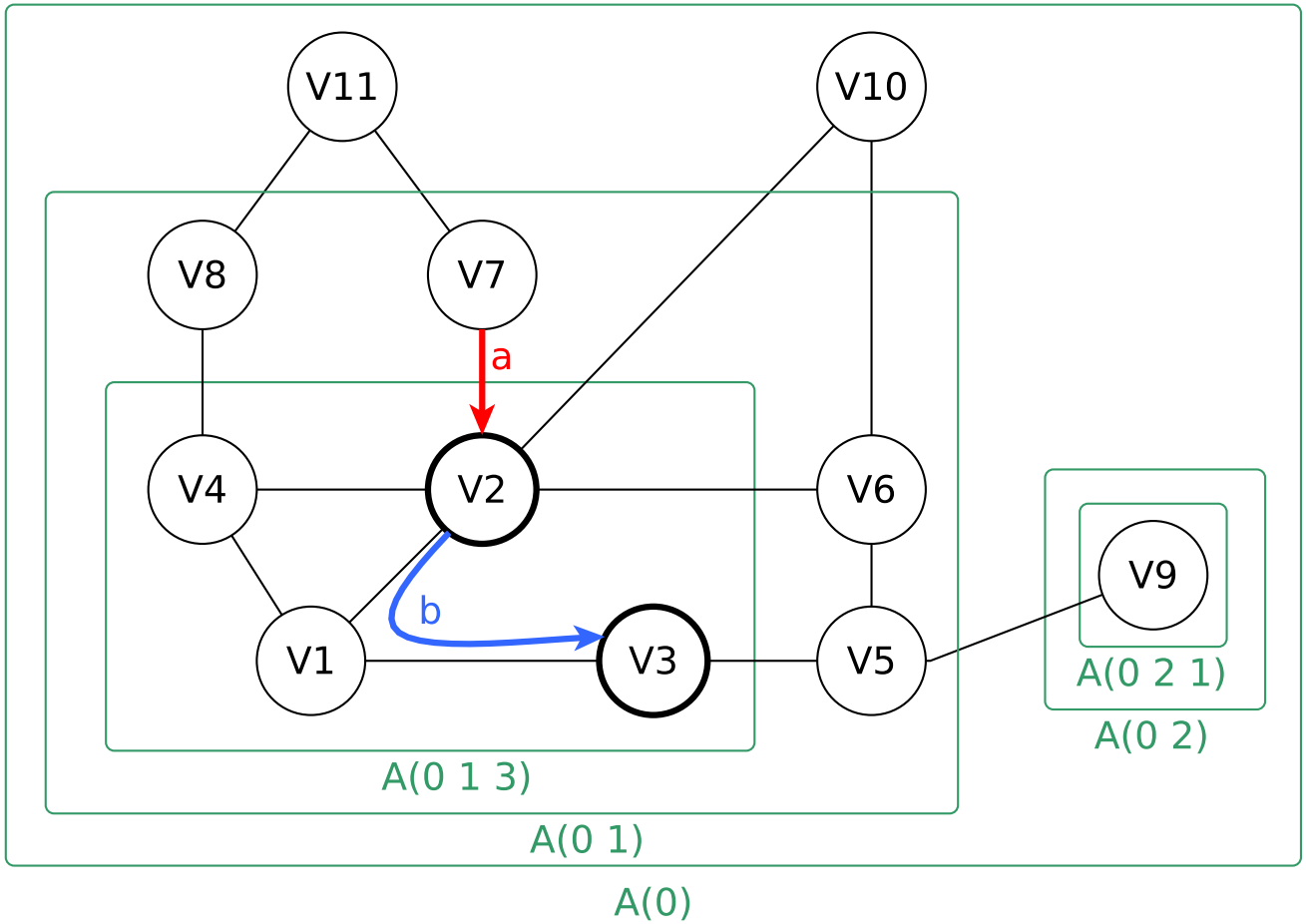
Network Dynamics: Example



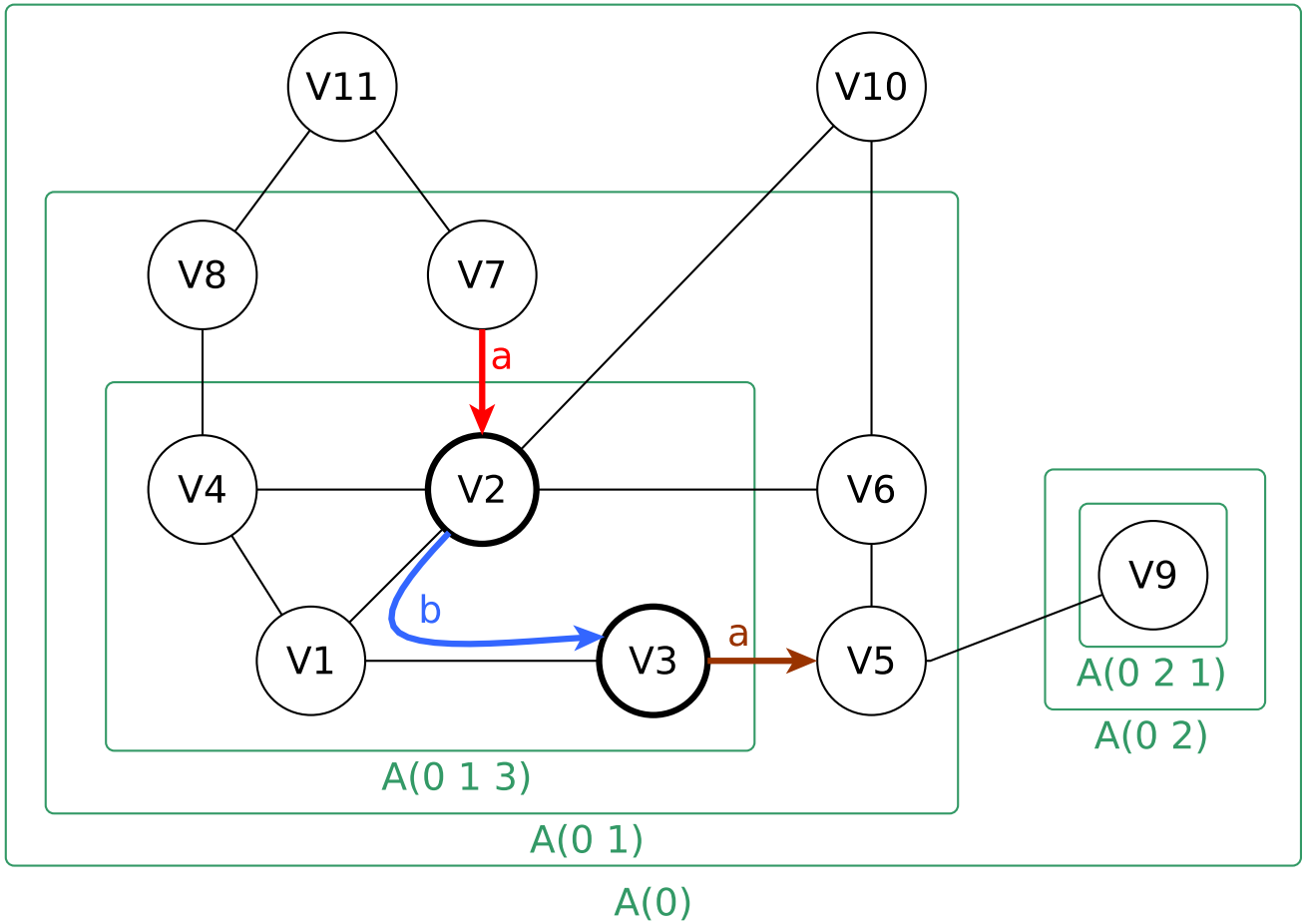
Network Dynamics: Example



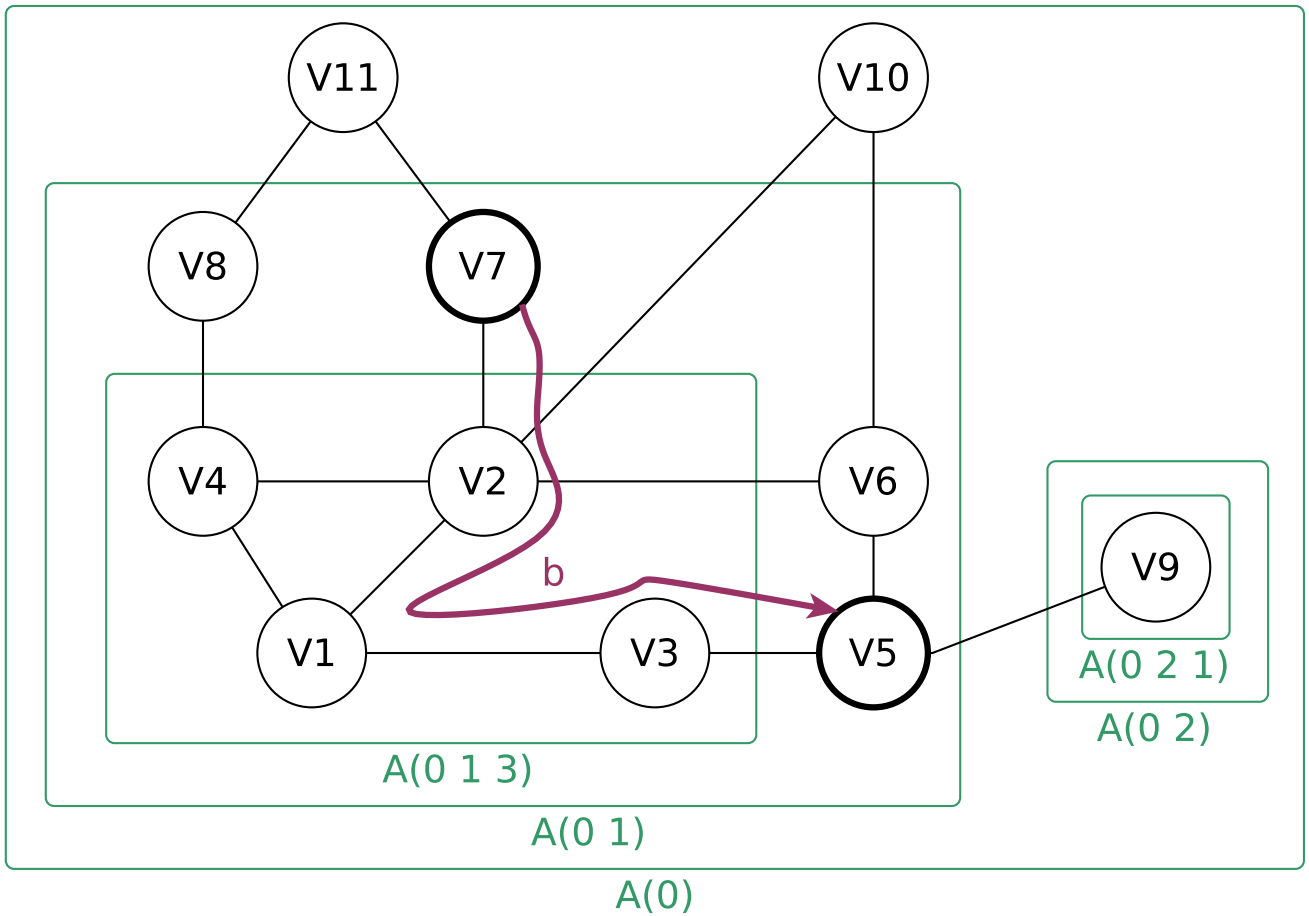
Network Dynamics: Example



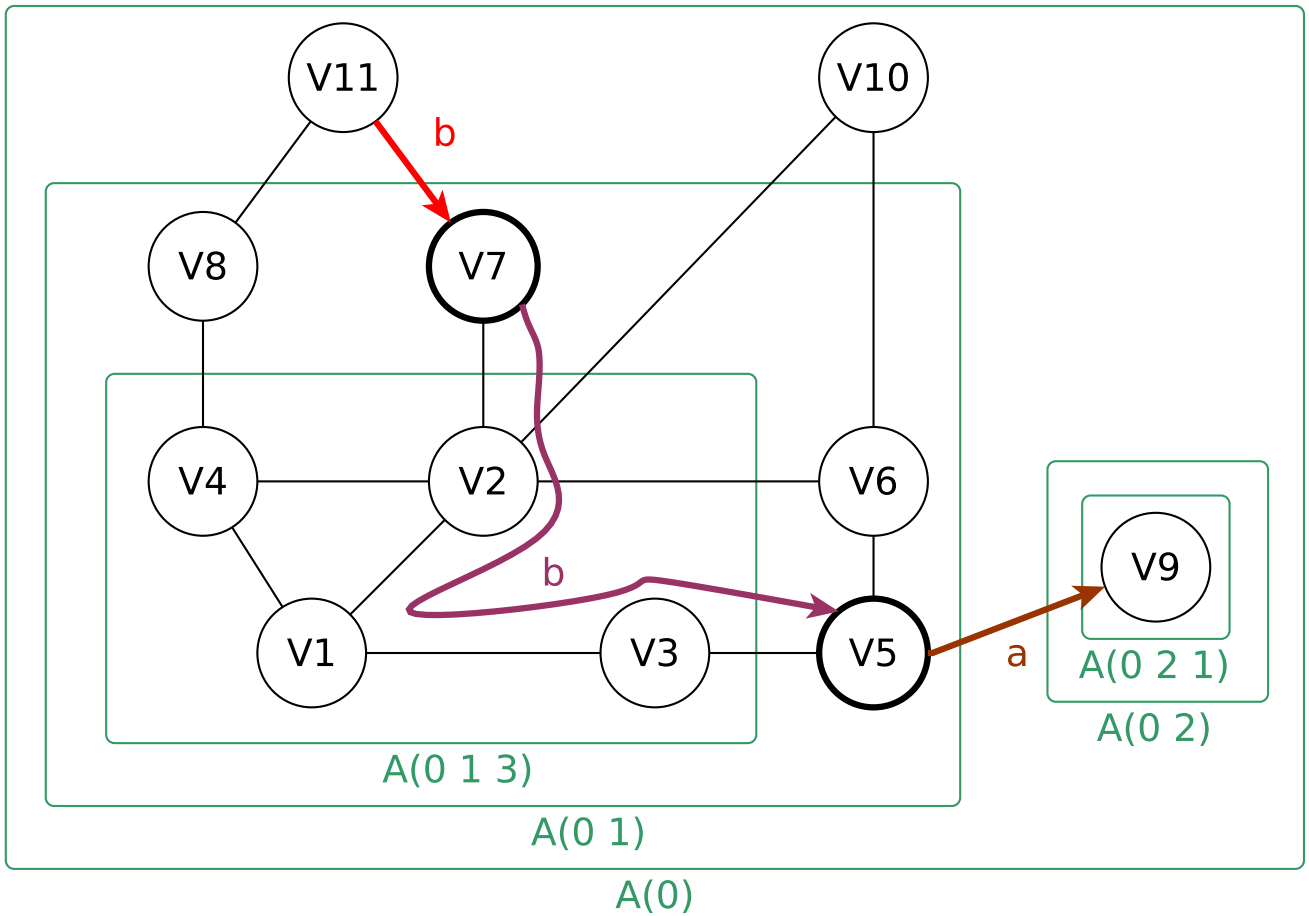
Network Dynamics: Example



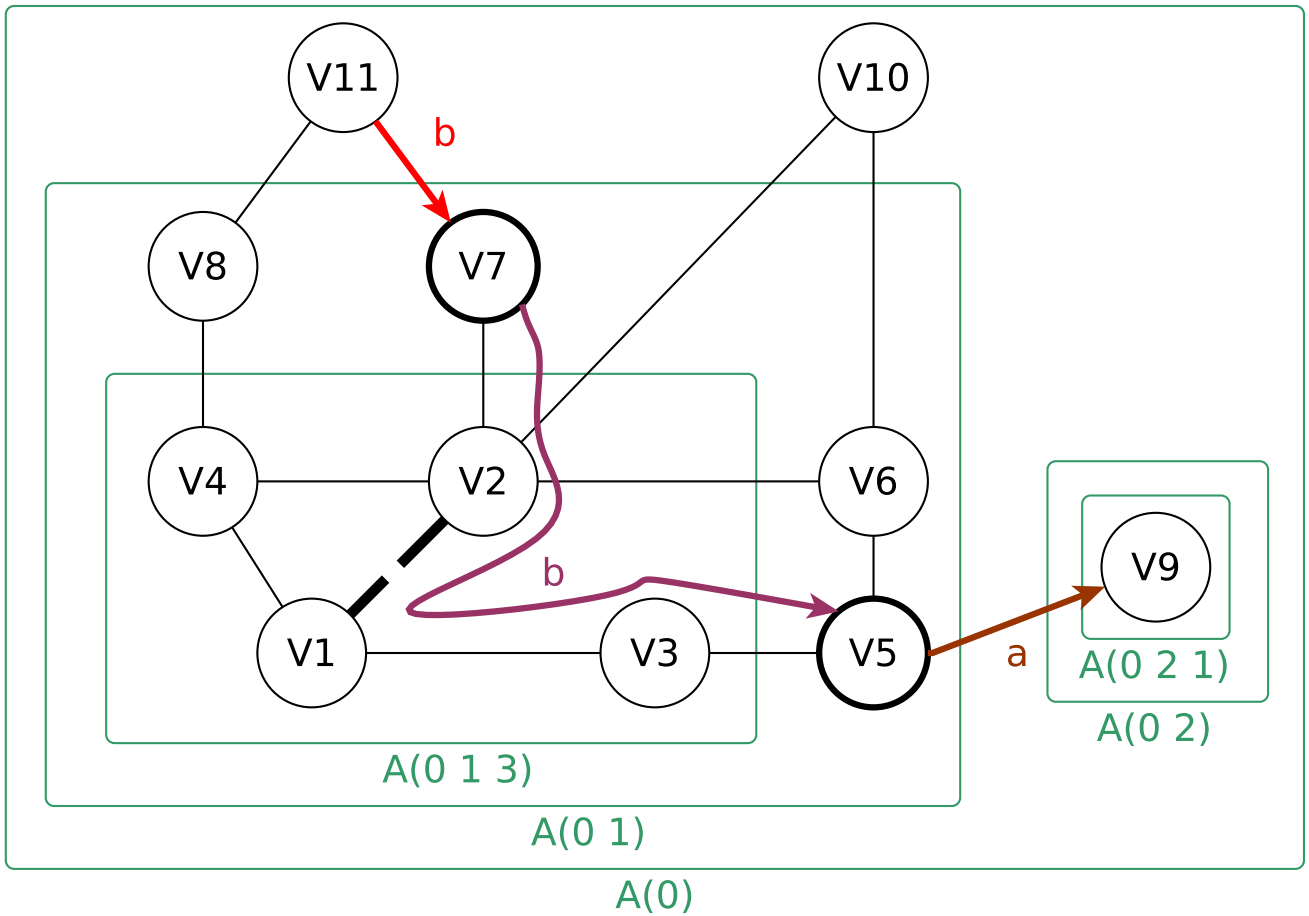
Network Dynamics: Example



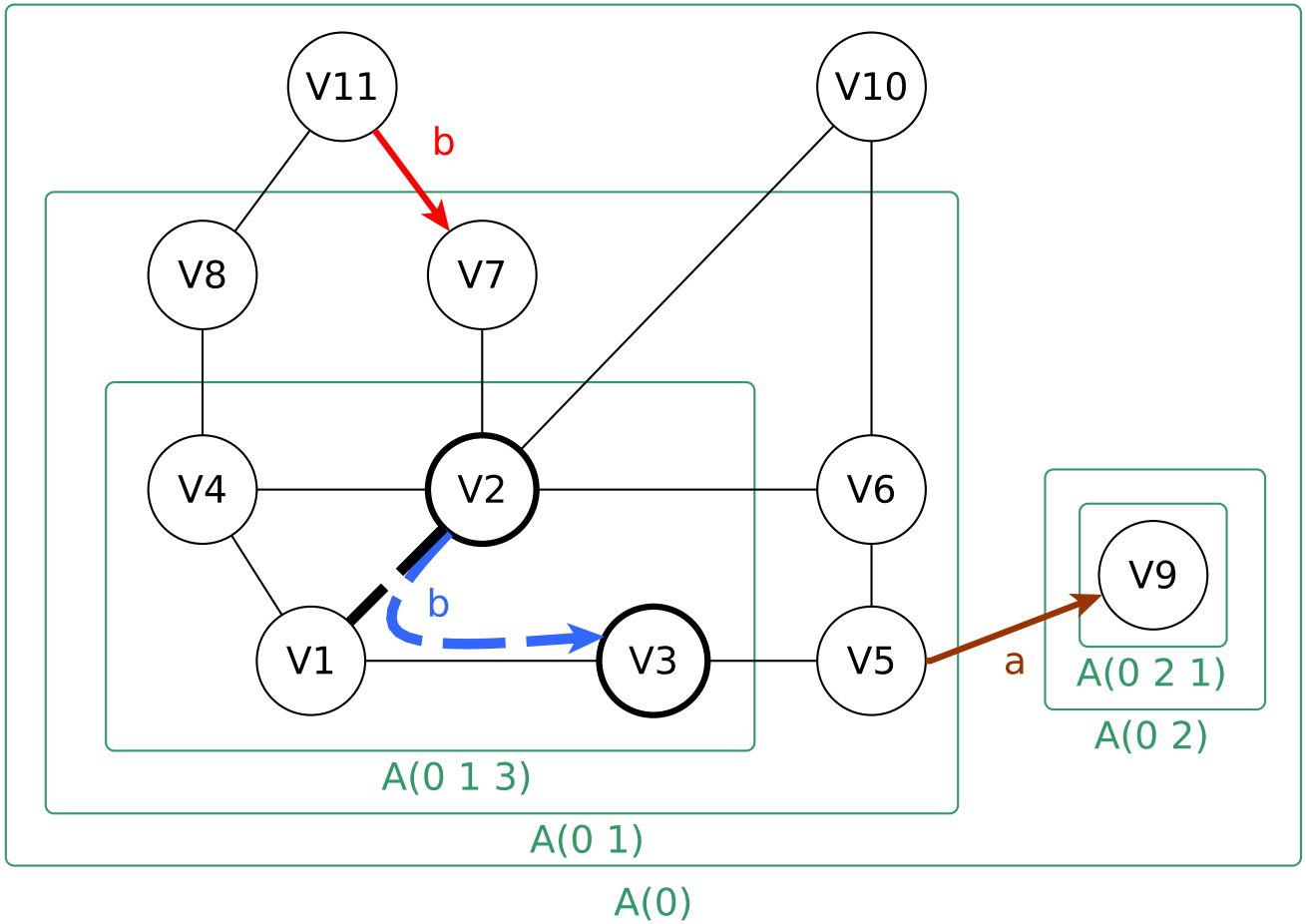
Network Dynamics: Example



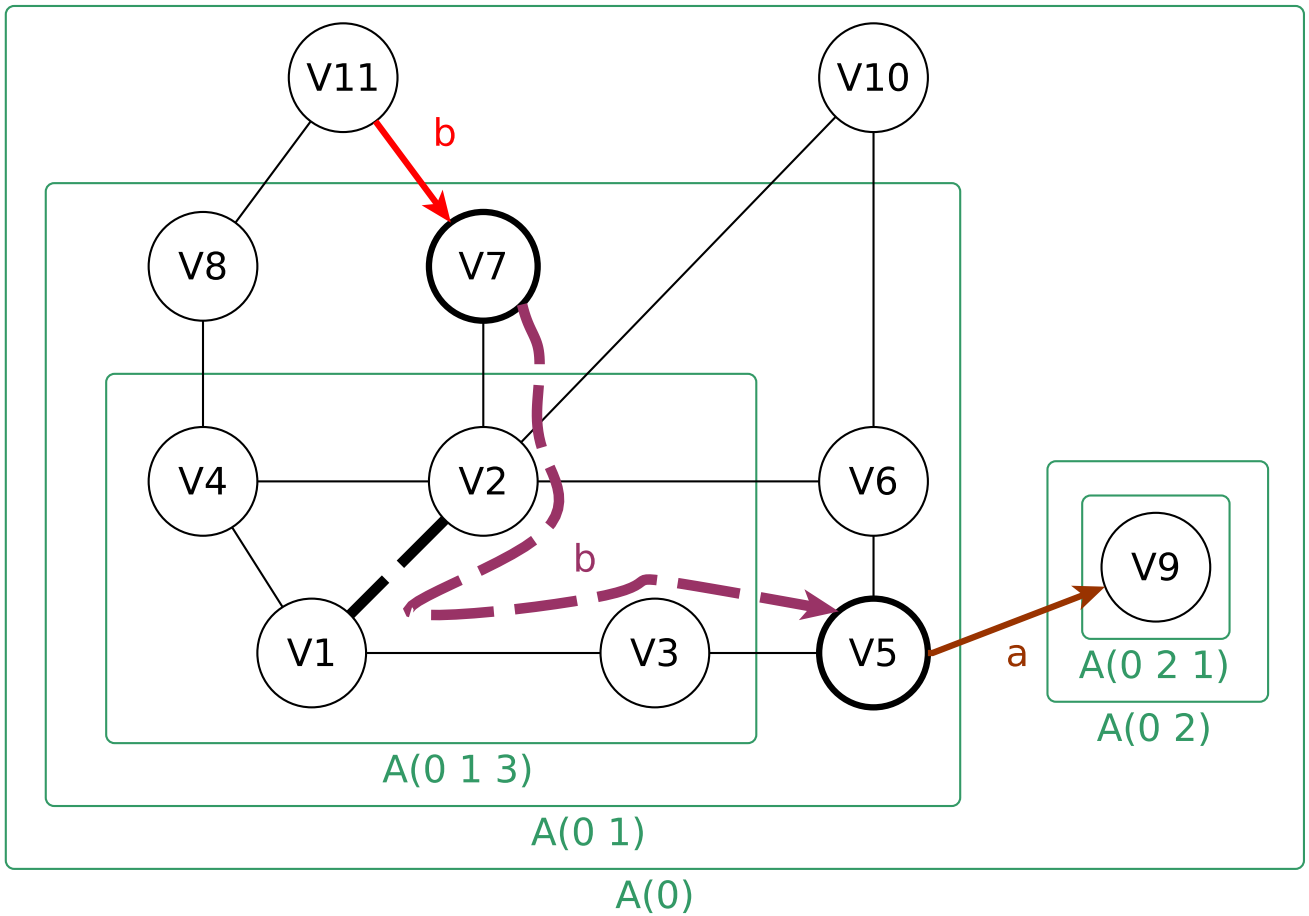
Network Dynamics: Example – Link Failure



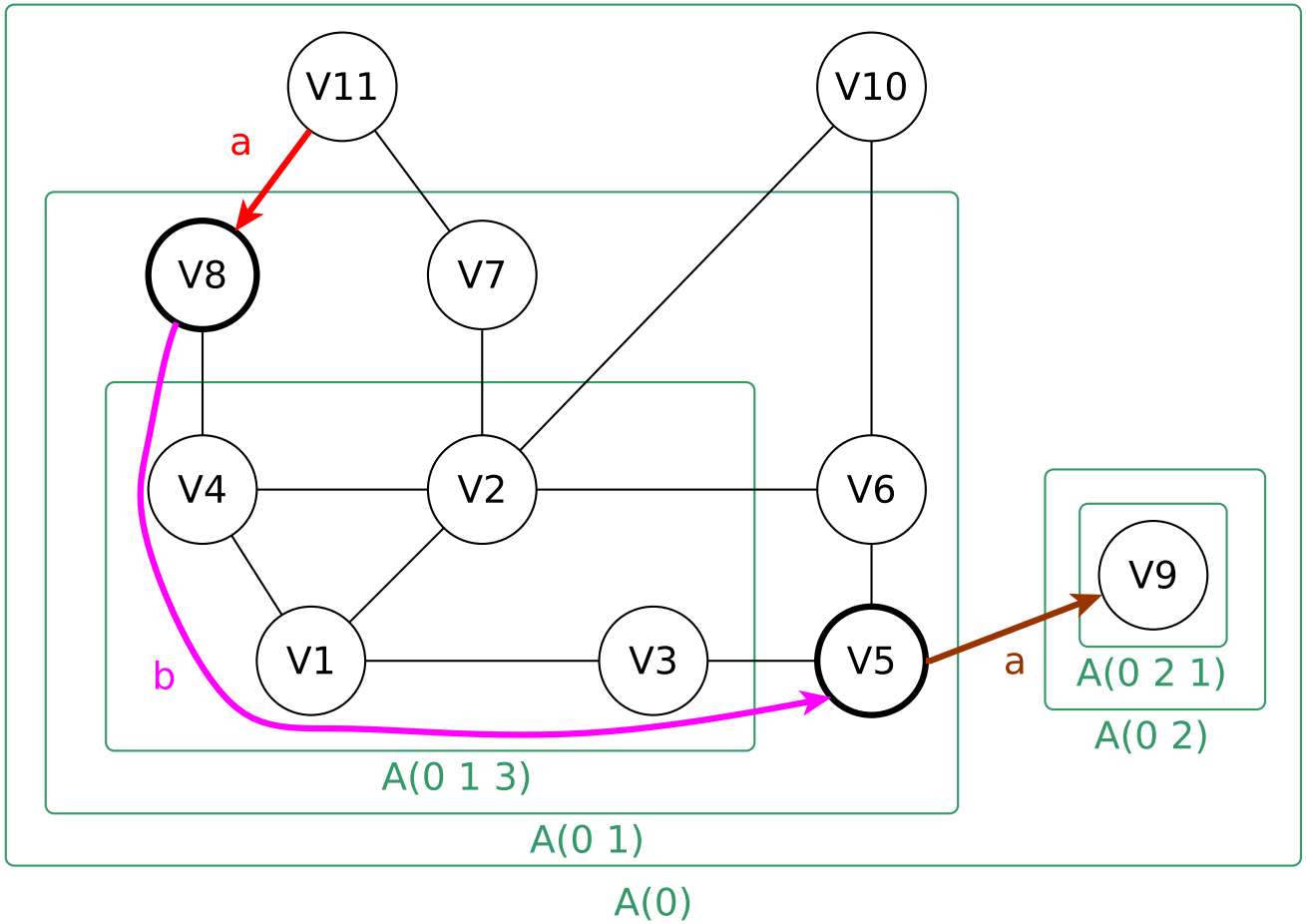
Network Dynamics: Example



Network Dynamics: Example



Network Dynamics: Example



Applicability Considerations

- Technologies
  - Our control plane is independent of the data plane that carries its messages
  - Messages are always exchanged between adjacent vertices, so only link-layer connectivity is required
  - Traffic forwarding is easy implementable with MPLS

Applicability Considerations

- Technologies
  - Our control plane is independent of the data plane that carries its messages
  - Messages are always exchanged between adjacent vertices, so only link-layer connectivity is required
  - Traffic forwarding is easy implementable with MPLS
- Quality of Service
  - Pathlets can be labeled with performance indicators
  - Pathlet composition rules may choose pathlets that fit the QoS requirements

Applicability Considerations

- Technologies
  - Our control plane is independent of the data plane that carries its messages
  - Messages are always exchanged between adjacent vertices, so only link-layer connectivity is required
  - Traffic forwarding is easy implementable with MPLS
- Quality of Service
  - Pathlets can be labeled with performance indicators
  - Pathlet composition rules may choose pathlets that fit the QoS requirements
- Similarities with Software Defined Networking and OpenFlow
  - Tight control on paths taken by flows
  - Hierarchical distribution of routing information (see, e.g., the controller hierarchy proposed in [Shimonishi et al., 2009])

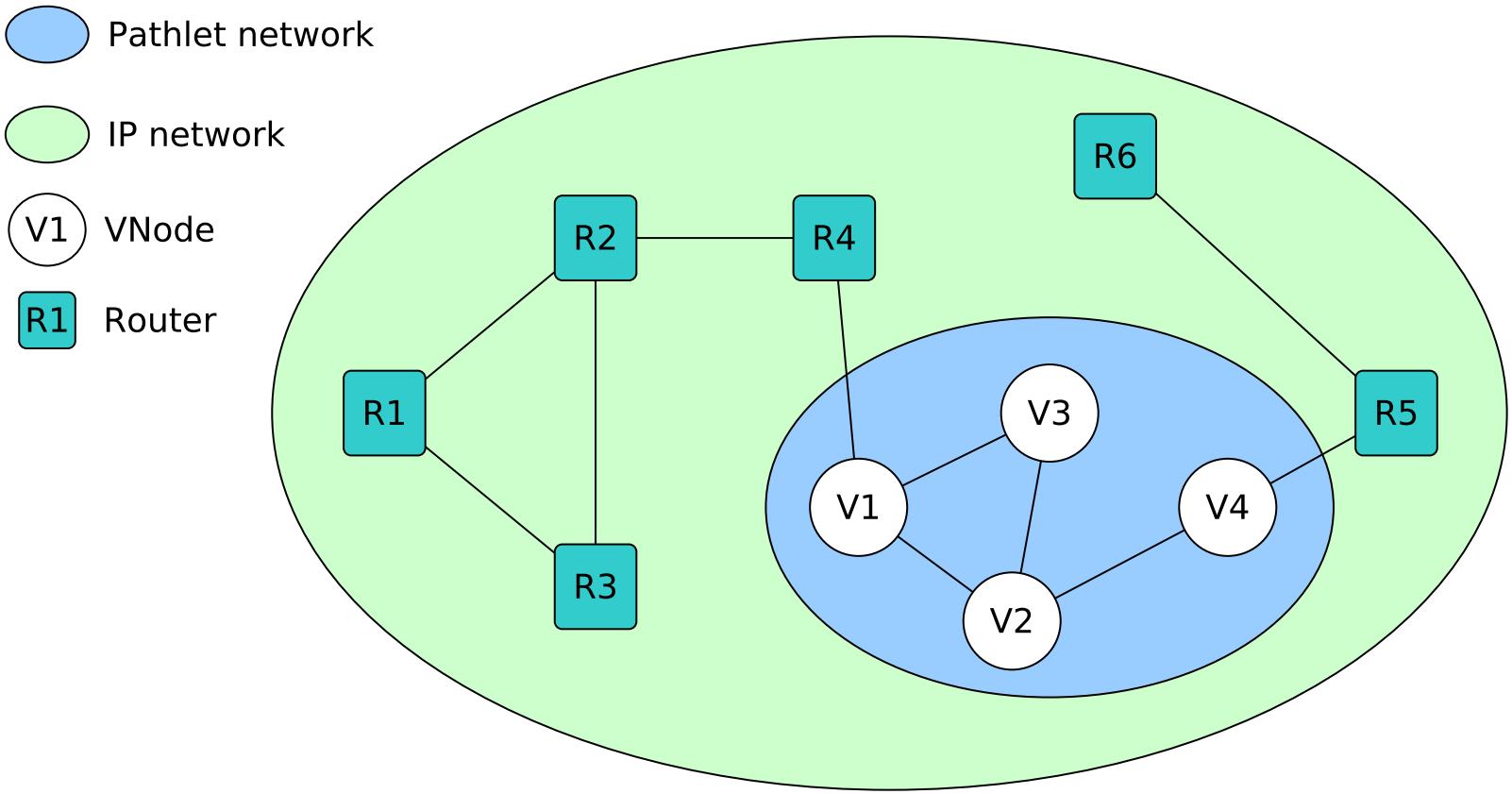
Incremental Deploy

- Intra-Domain Pathlet Routing is designed to support an incremental deploy

Incremental Deploy

- Intra-Domain Pathlet Routing is designed to support an incremental deploy
- We consider two main scenarios:
  - A *Pathlet-network* is embedded in an *IP-network*
  - A *Pathlet-network* contains a "legacy" *IP-network*

IP-network into Pathlet-network



Experimental Evaluation

- We implemented a prototype using OmNET++ simulator
- We tested the functionality and scalability of our approach on topologies created using an ad-hoc generator
  - Some input parameters: number of router per area, number of areas in each area, length of the label stacks, etc.
- Our experiments involve:
  - Variation of number of the areas in each area in the range $[2, 7]$
  - Variation of length of label stack in the range $[1, 4]$

Results

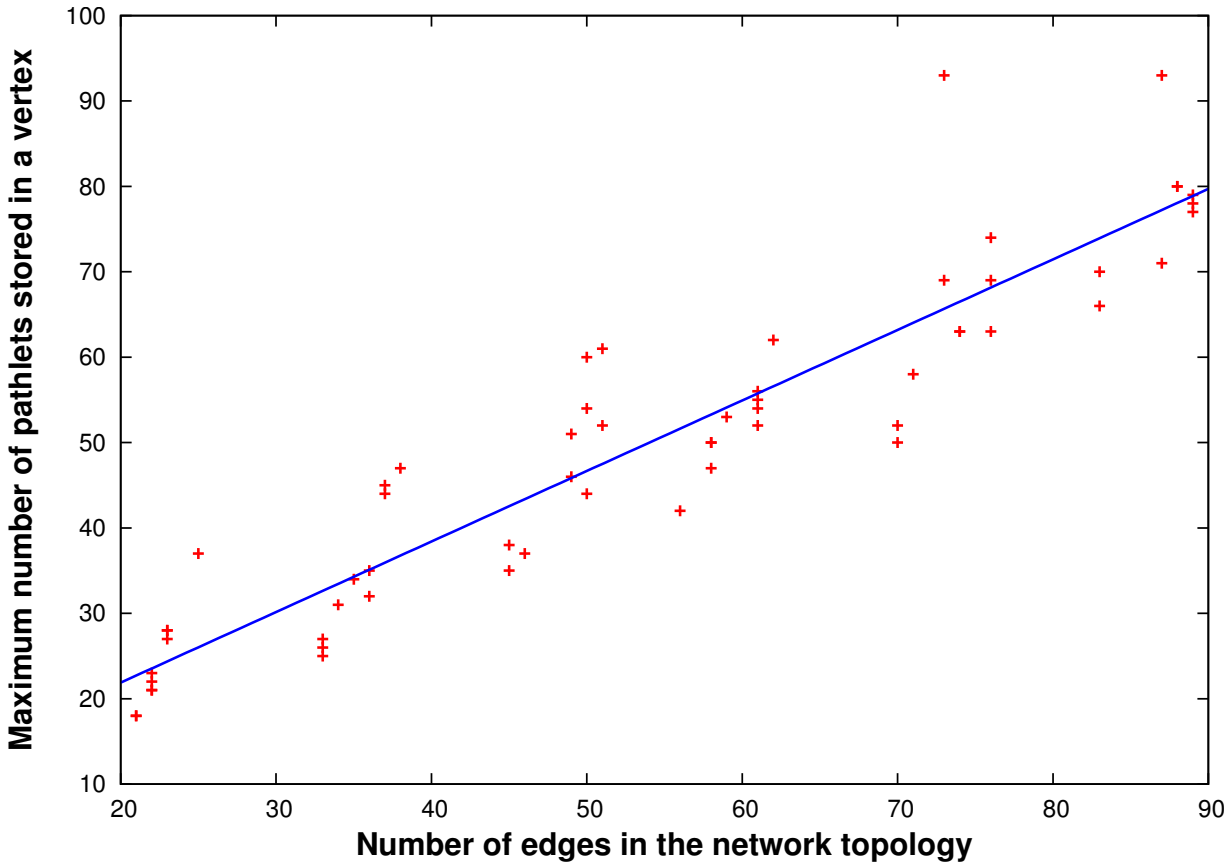


Figure : Maximum number of pathlets stored in a router

Results

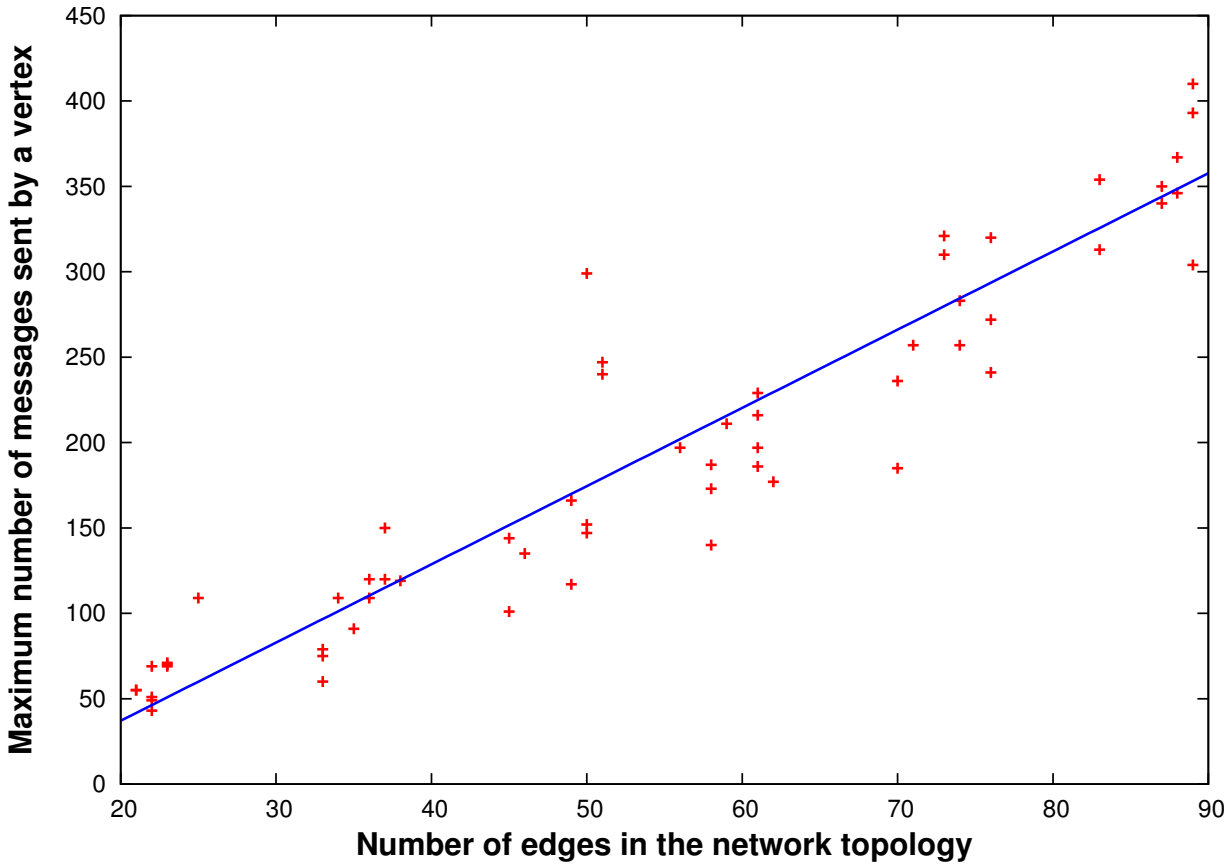


Figure : Maximum number of messages sent by a router

Convergence Time

- We set link delays to random values, with a uniform distribution in the range between 10ms and 50 ms
- We used pathlet a pathlet composition rule that composes all possible pathlets

Convergence Time

- We set link delays to random values, with a uniform distribution in the range between 10ms and 50 ms
- We used pathlet a pathlet composition rule that composes all possible pathlets
- We observed that convergence times are below 1s:
  - $[363ms, 611ms]$, changing the number of areas in each area
  - $[259ms, 736ms]$, changing the length of the label stack

Convergence Time

- We set link delays to random values, with a uniform distribution in the range between 10ms and 50 ms
- We used pathlet a pathlet composition rule that composes all possible pathlets
- We observed that convergence times are below 1s:
  - $[363ms, 611ms]$, changing the number of areas in each area
  - $[259ms, 736ms]$, changing the length of the label stack
- We believe these results are promising in terms of scalability

Conclusions and Future Works

- We introduce a control plane that has several interesting features:
  - Control of routing paths at different levels of granularity
  - Scalability
  - Resiliency
  - Support for Quality of Service
- We open new research perspectives:
  - Refinement of pathlet composition rules to support further requirements
  - Improving the behavior of vertices that purge unusable pathlets due to routing policies
  - Mechanism to transparently replace pathlets, due to administrative configuration changes
  - Improve the handling of faults

Thank you!