SSFNet and Routing Simulation

BJ Premore

Dartmouth College

Overview

- What is SSFNet?
- Building a Model
- Protocol Packages
- A Routing Study

What is SSFNet?

- Modern software for modeling and simulation of large networks
- Java-based
- IP packet-level granularity

Features

- Fully Integrated Network Environment
  - many detailed network components included
  - components all inter-operable
- Scalability
  - designed to handle large, complex simulations
  - achievable simulation sizes vary by model and hardware
- Configurability
  - all components have multiple configurable attributes
  - sometimes above and beyond actual implementations
- And more ...
  - repeatability
  - random number package (CERN Colt)
  - plotting
  - monitoring

SSFNet Architecture

DML = Domain Modeling Language

- model configuration

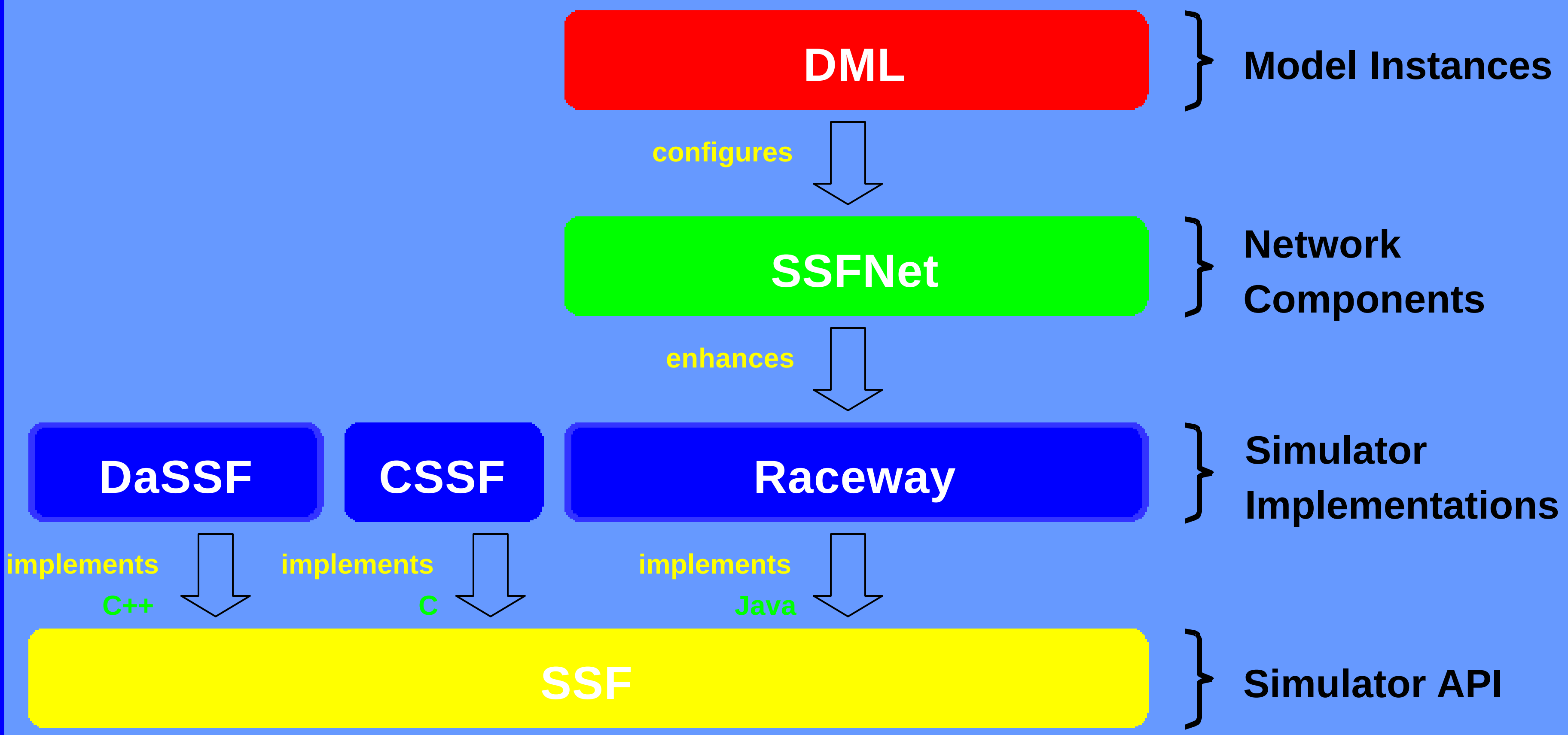
SSFNet = SSF Network Models

- not independent

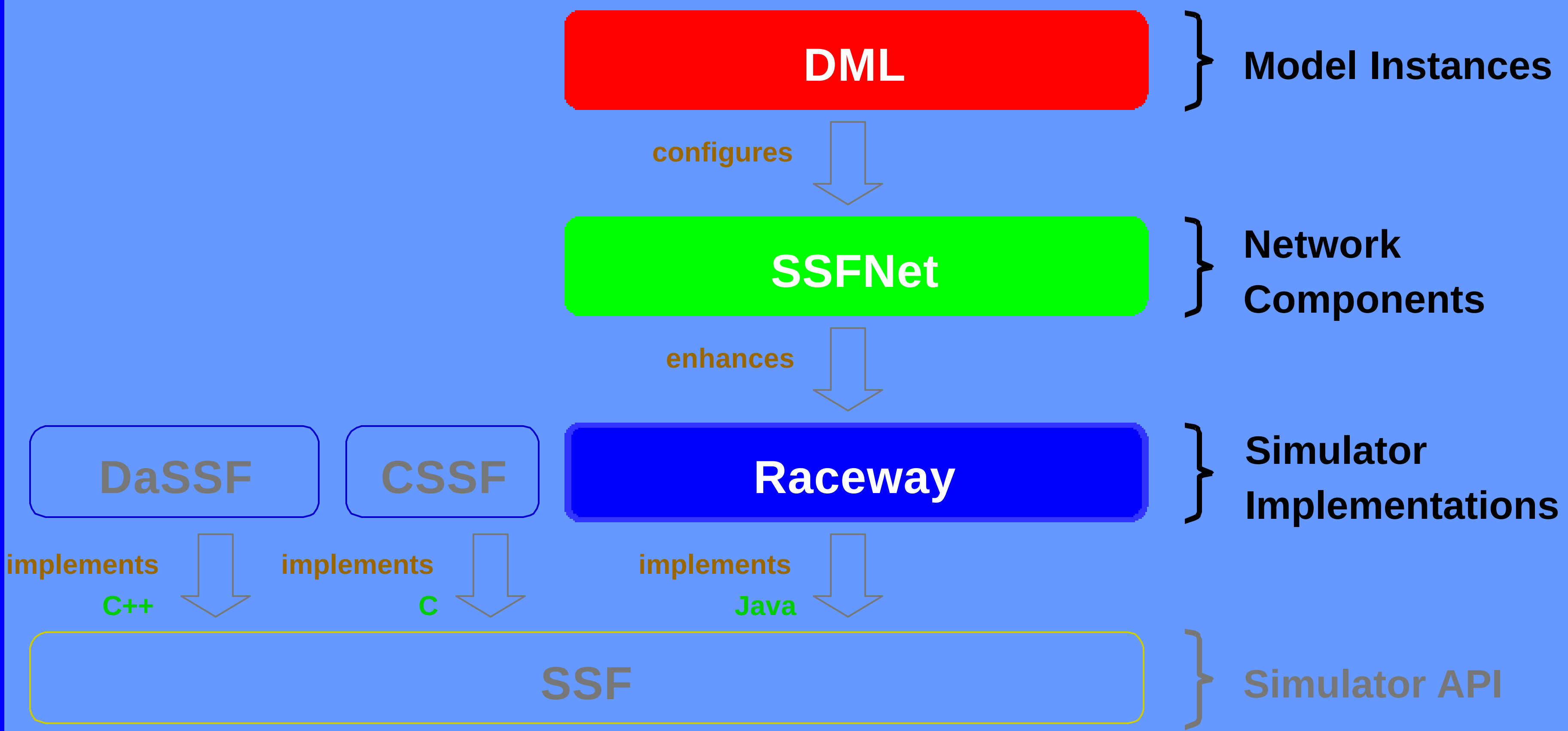
SSF = Scalable Simulation Framework

- a standard for discrete-event simulation of large, complex systems

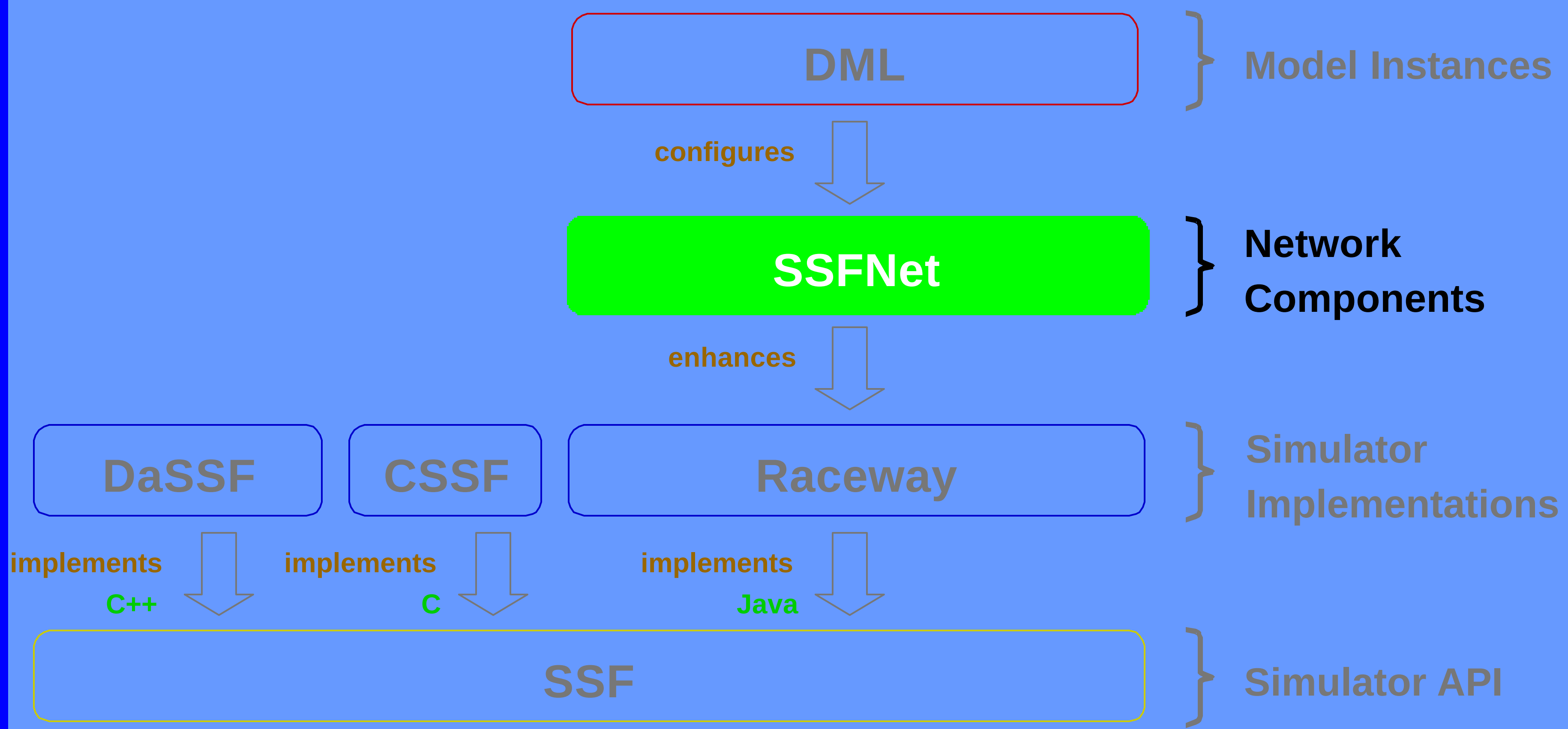
Simulation Layers



IDE Package


DML
Model Instances
configures
SSFNet
Network Components
enhances
DaSSF
CSSF
Raceway
Simulator Implementations
implements
C++
implements
C
implements
Java
SSF
Simulator API

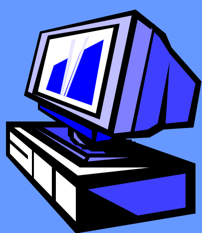
SSFNet Layer



Network Components

- each is a Java class or Java package
- includes state, behavior, config info

physical entities

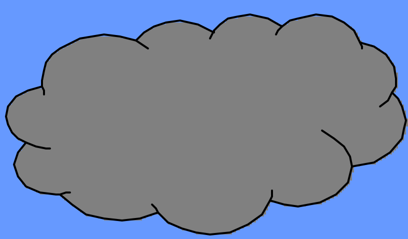


host

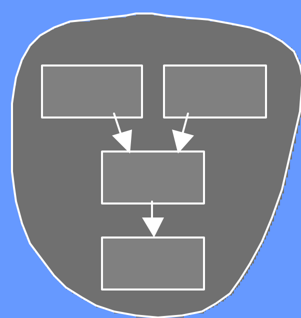
router

link

logical containers



Net



protocol graph

protocols

IP

TCP

Sockets

FTP client

BGP

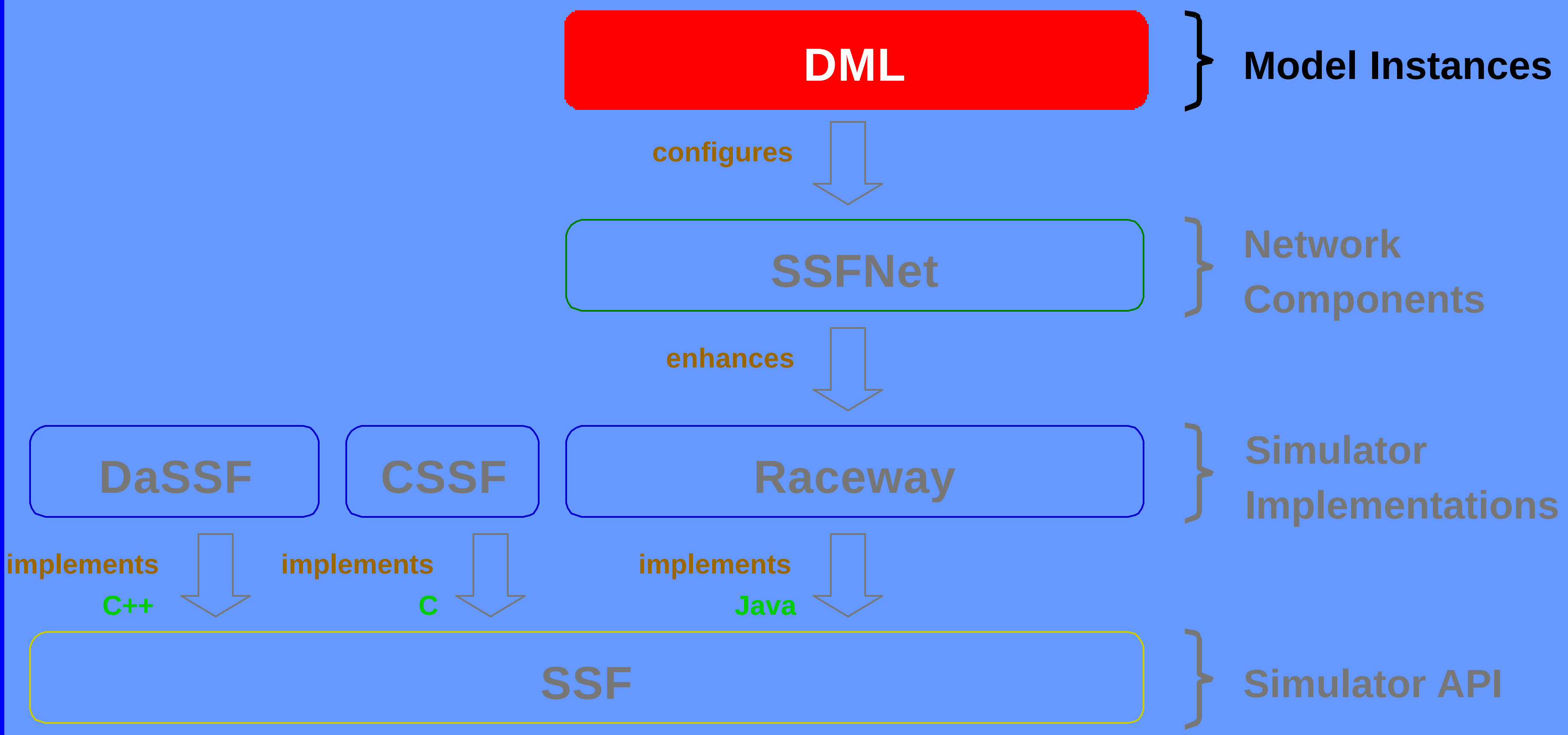
HTTP client

OSPF

Building a Model

- think hierarchically
- understand NHI addressing

DML Layer


DML
Model Instances
configures
SSFNet
Network Components
enhances
DaSSF
CSSF
Raceway
Simulator Implementations
implements
C++
implements
C
implements
Java
SSF
Simulator API

Basic DML Properties

- goal: simplicity
- attribute/value pairs
- hierarchical
- extensibility
- substitution

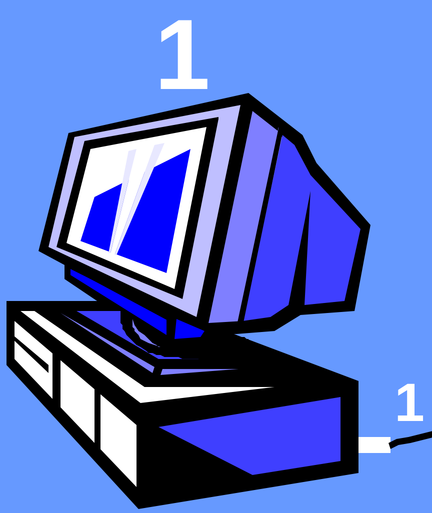
Basic DML Examples

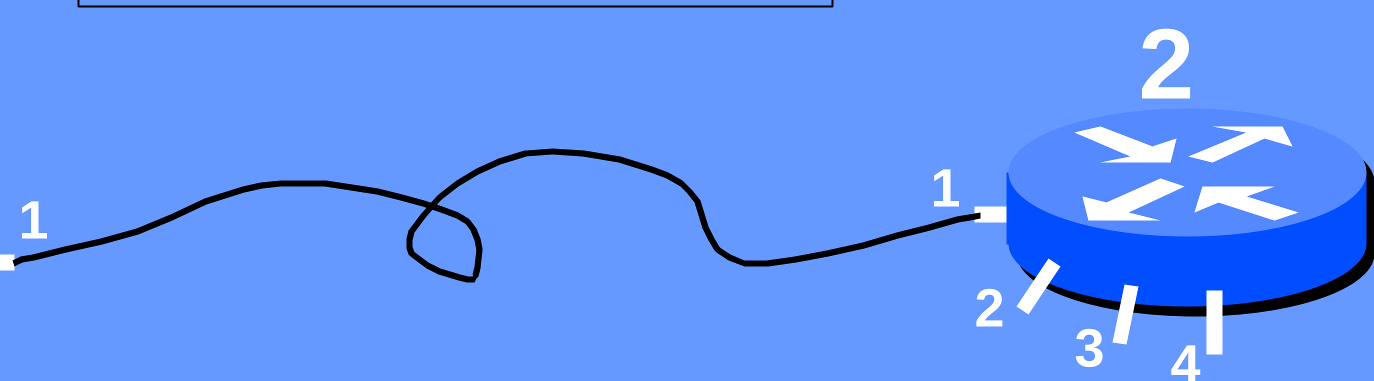
- attribute/value pairs
  - simple attributes
    - cat Morris
    - bandwidth 1.544Mb
    - random_string “a1b 2$#[_4bs”
  - nested attributes
    - cat [name Morris]
    - cat [name Morris age “10 years”]

```
cat [
  name Morris
  age “10 years”
  color [
    primary orange
    secondary white
    pattern stripes
  ]
]
```

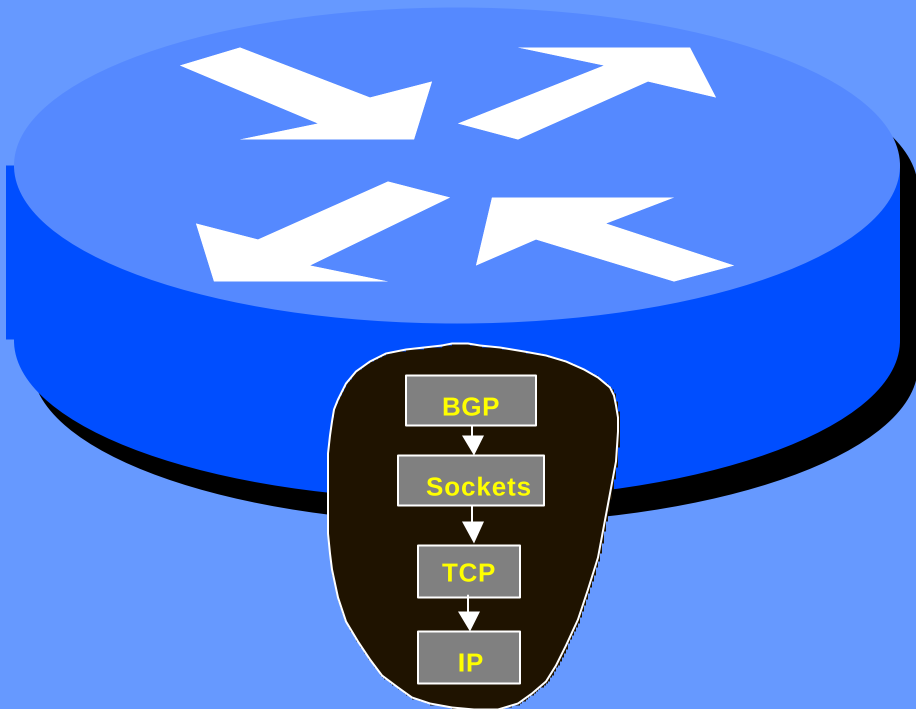
DML Example

```
host [
  id 1
  interface [ id 1 ]
]
router [
  id 2
  interface [ idrange [ from 1 to 4 ] ]
]
link [
  attach 1(1)
  attach 2(1)
]
```





DML: The Protocol Stack



protocol graph

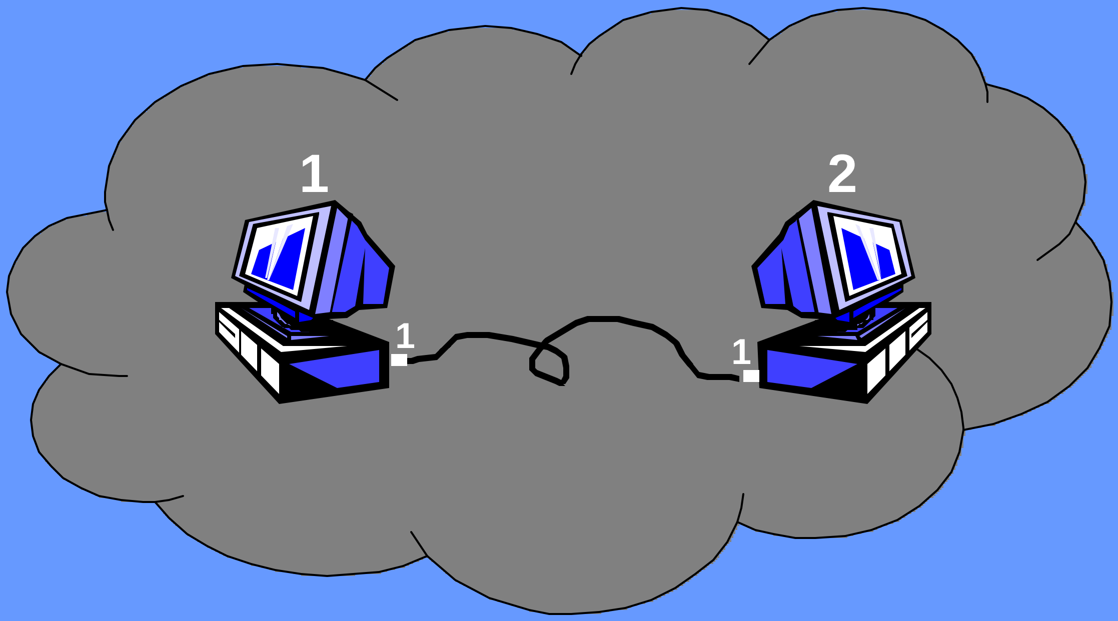
```
router [
  graph [
    ProtocolSession [
      name bgp
      use SSF.OS.BGP4.BGPSession
    ]
    ProtocolSession [
      name ospf
      use SSF.OS.OSPF.sOSPF
    ]
    ProtocolSession [
      name tcp
      use SSF.OS.TCP.tcpSessionMaster
    ]
    ProtocolSession [
      name ip
      use SSF.OS.IP
    ]
  ]
]
```

NHI Addressing

- Internal format for model-building convenience
- N:N:N: ... :N:H(I)
  - N = network id
  - H = host id
  - I = interface id
- top-level Net cannot have id
- local vs. global
  - local link need not attach to global NHI address
  - networks and hosts may be abbreviated or omitted

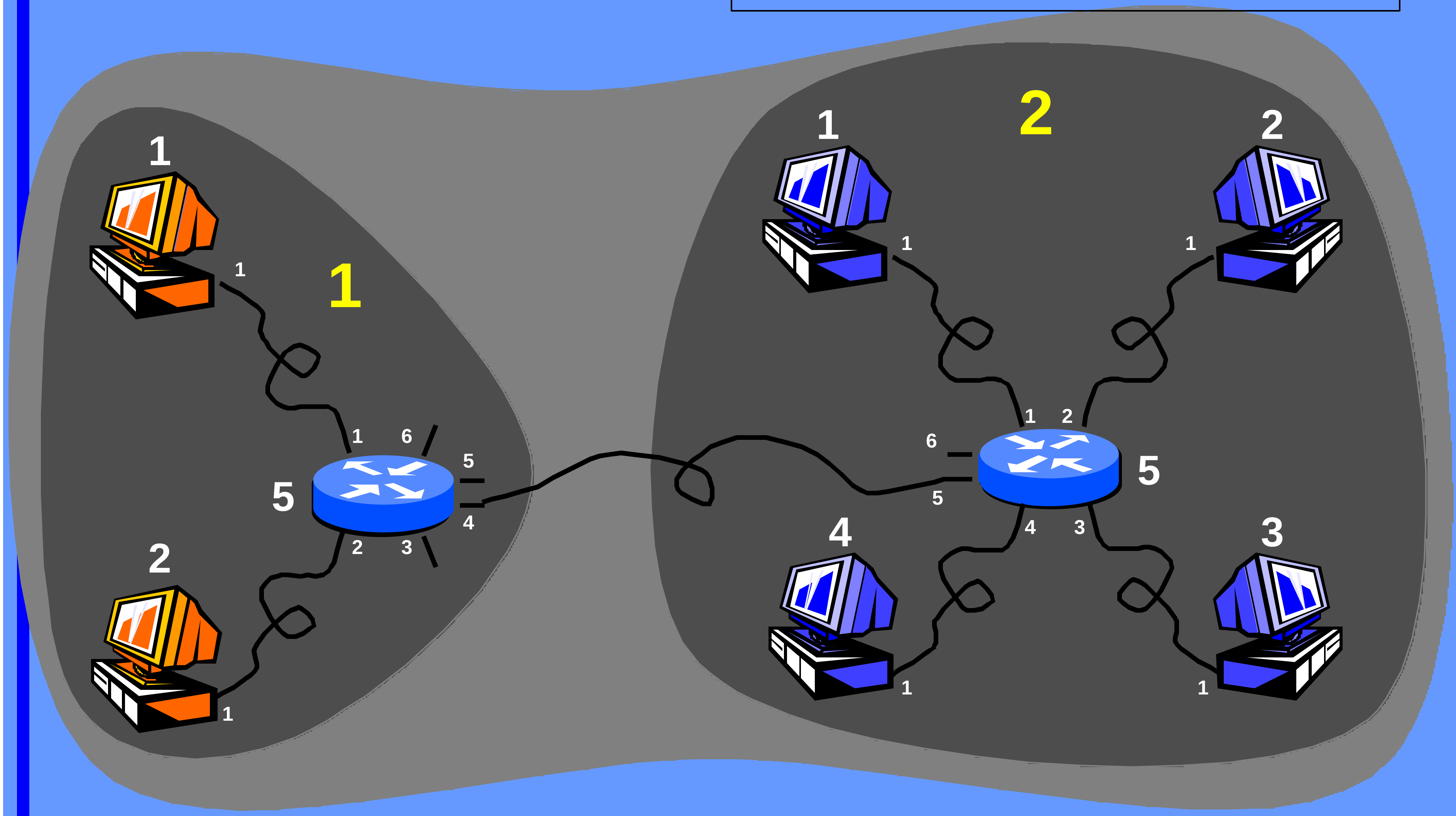
Hierarchy Example

```
Net [
  host [ id 1  interface [ id 1 ] ]
  host [ id 2  interface [ id 1 ] ]
  link [ attach 1(1) attach 2(1) ]
]
```



Hierarchy Example 2

```
Net [
  Net [ id 1 ... ] # 2 hosts + 1 router
  Net [ id 2 ... ] # 4 hosts + 1 router
  link [ attach 1:5(4) attach 2:5(5) ]
]
```



From Installation to Execution

- download distribution from www.ssfnet.org
- unzip in location of your choice
- set CLASSPATH environment variable
- while in ssfnet/ directory, type make
  - builds and validates
- use favorite editor to create DML model
- execute it: java SSF.Net.Net *runtime dml-file*

SSFNet Protocol Models

- IP (simplified)
- TCP (validated)
- UDP
- Sockets
- OSPF (two versions)
- BGP
- HTTP and FTP clients
- Widgets

Applications

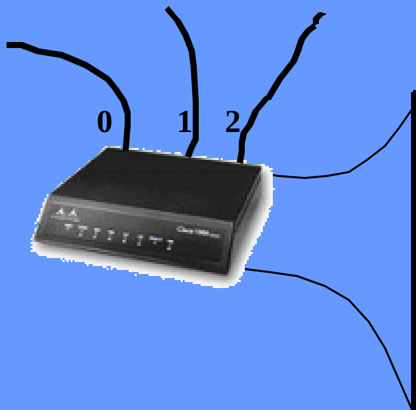
- IPsec, MPLS at NIST
- SNMP and NFS client/server at SHAI
- BGP route flap dampening
- in university courses

A Routing Study

- a general inquiry into routing dynamics
- not as well-understood as other protocols
  - distributed behavior more complex than end-to-end
- some parameters pulled "out of the blue sky"
- ubiquitously used in Internet
- using the BGP model in SSFNet
  - full-fledged routing models not previously available

Routing Basics

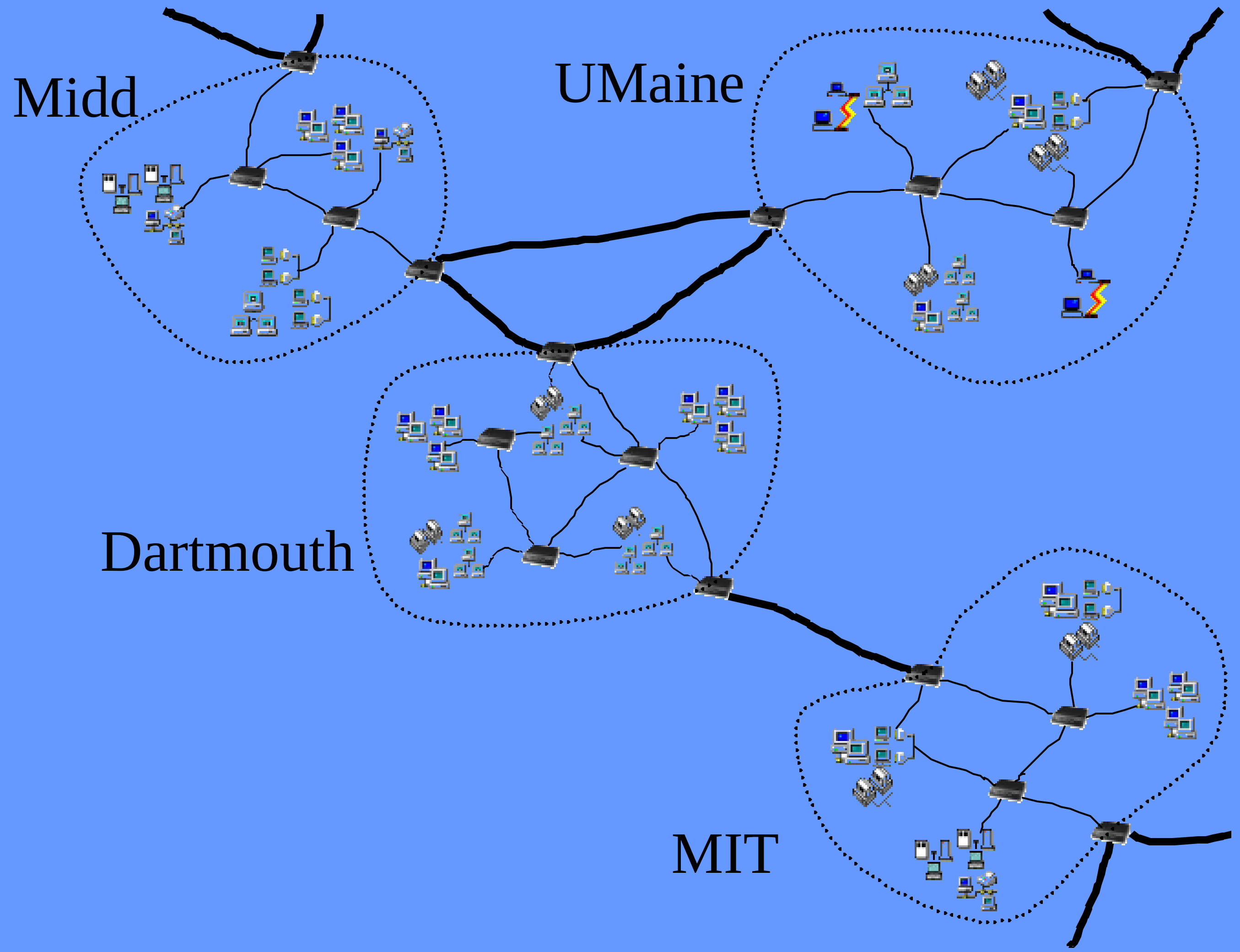
- forwarding vs. routing



Destination	Direction
129.17.29.32	1
192.14.155.13	0
129.44.100.1	2
128.42.0.1	1

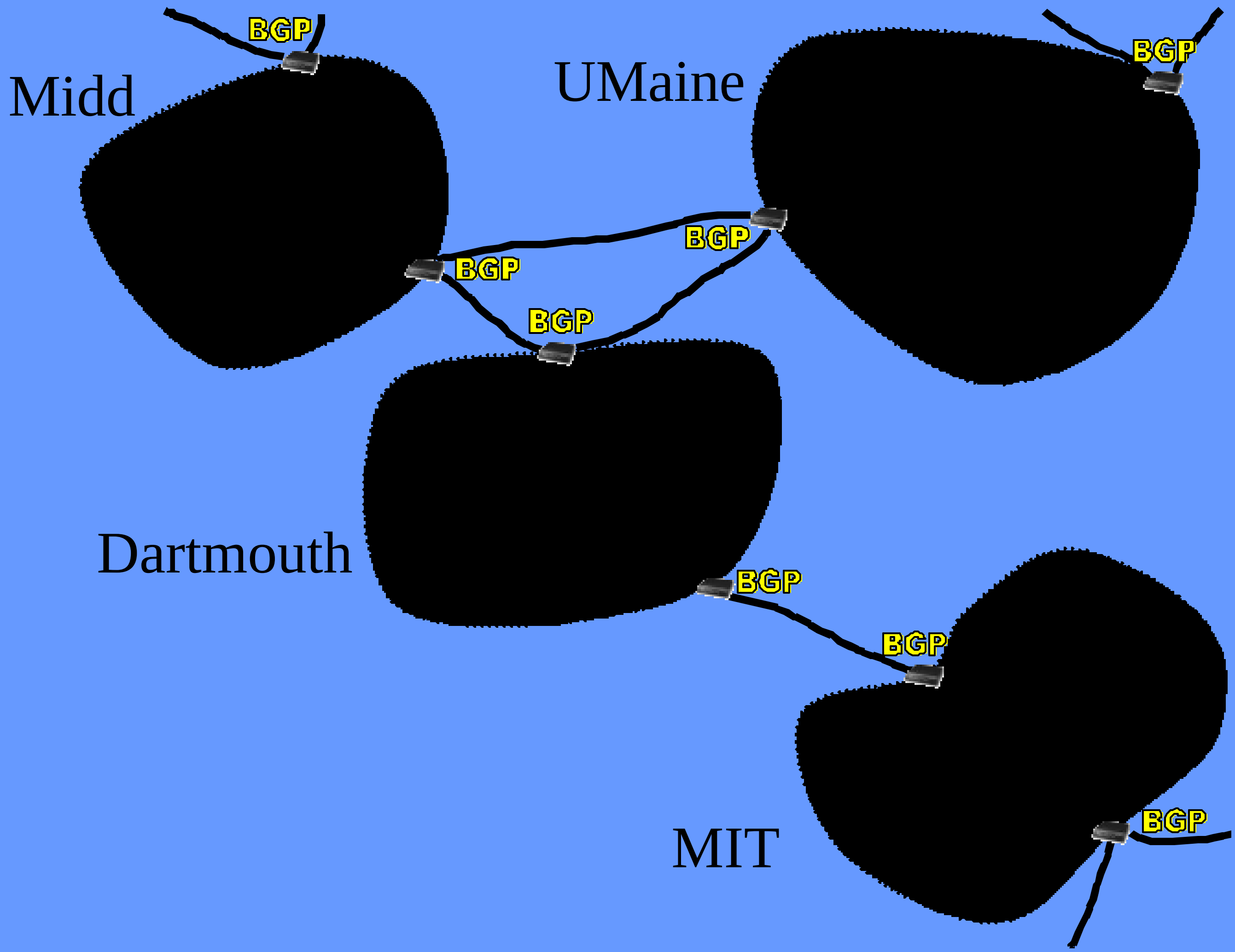
Routing Basics

- Two-level routing hierarchy



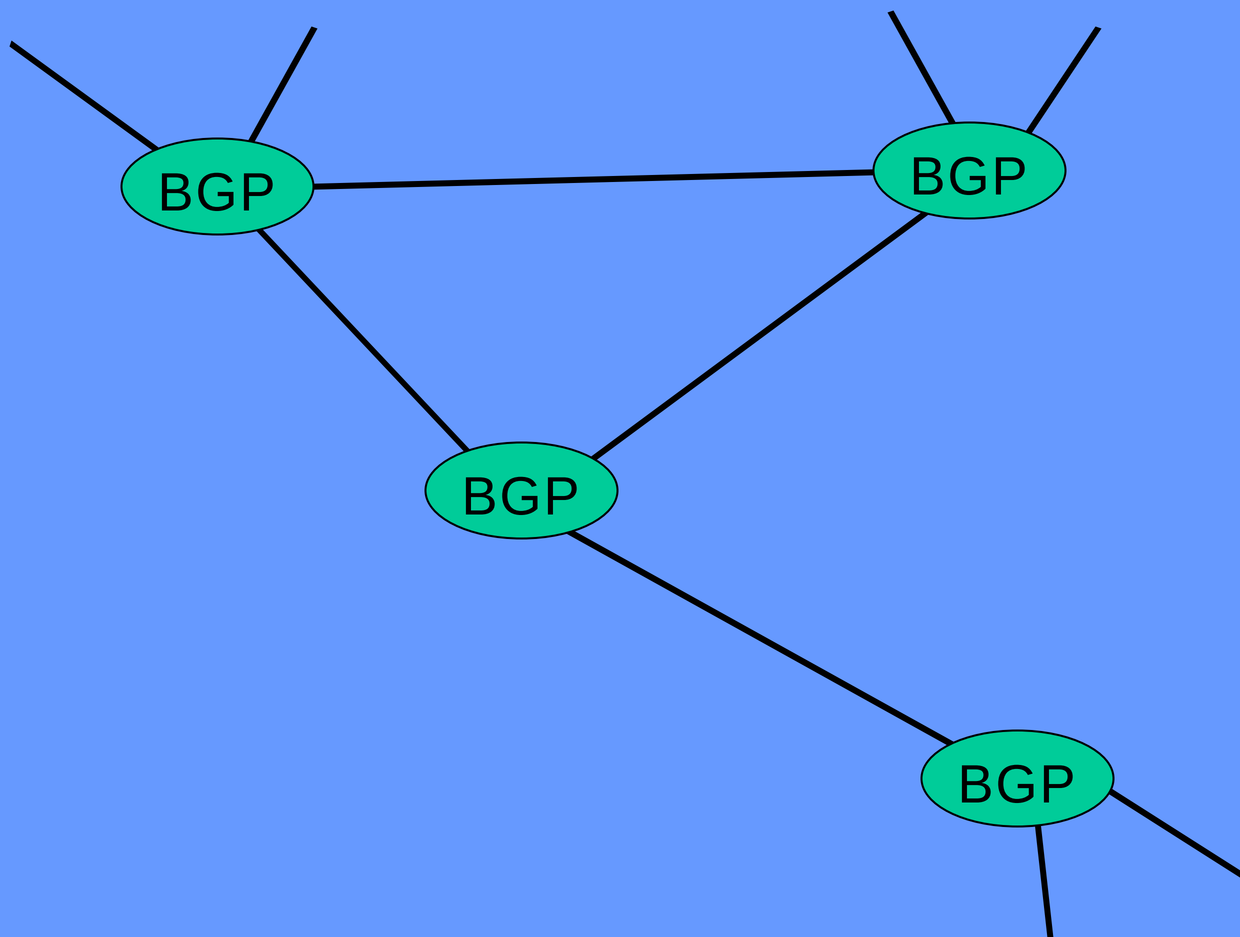
Routing Basics

- BGP used for inter-domain routing



Routing Basics

- a simplified view



What is BGP?

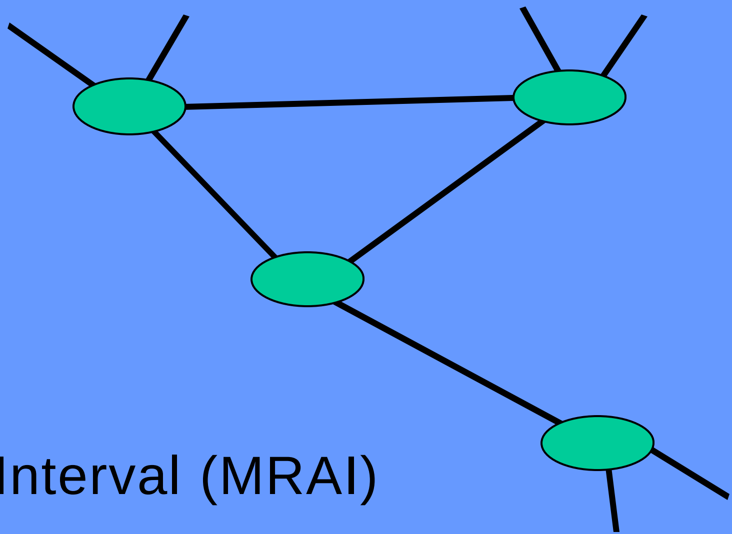
- BGP is a distributed all-points preferred path algorithm, essentially
- the glue that holds the Internet together

BGP Basics

- routing algorithm
  - – 1. Learn neighbors
  - – 2. Share reachability information with neighbors
  - – 3. Continue sharing updated reachability information
- incremental updates
  - advertisements
  - withdrawals

may serve as *implicit* withdrawals

- decisions
  - neighbors paths + policy
- rate limiting
  - Minimum Route Advertisement Interval (MRAI)



SSFNet BGP

- Based on RFCs
  - RFC 1771: BGP-4 and latest drafts
  - RFC compliant implementation
  - Includes some RFC-specified extensions (Route Reflection)
  - Has features similar to those used by vendors (policy-based filtering)

SSF.OS.BGP4 Functionality

- Finite state machine, timers, RIB
- TCP transport
- Peering: exterior and interior
  - Route reflection
- Messages and path attributes
- Policy
  - filter based on path attribute
  - attribute modification
- Monitoring of protocol operation
  - gather stats on practically any event of interest

Validation Methodology

- No standards, create our own suite
- Basic behavior in simple topologies
  - Peering session maintenance (Hold & KeepAlive timer operation)
  - Route advertisement and withdrawal
  - Route selection
  - Reflection
  - Internal BGP
- General behavior in complex topologies
  - End-to-end data delivery
  - Exercises basic behaviors as well
- Policy testing
  - Converging and non-converging gadgets [Griffin 1999]

BGP Convergence

- Given a change in the network, how long does it take for all BGP speakers to return to a stable state?

Previous Work

- no convergence bound
  - persistent oscillations possible

    [Varadhan, Govindan, Estrin 1997]
- empirical measurements
  - lots of updates!
  - convergence not so good ...

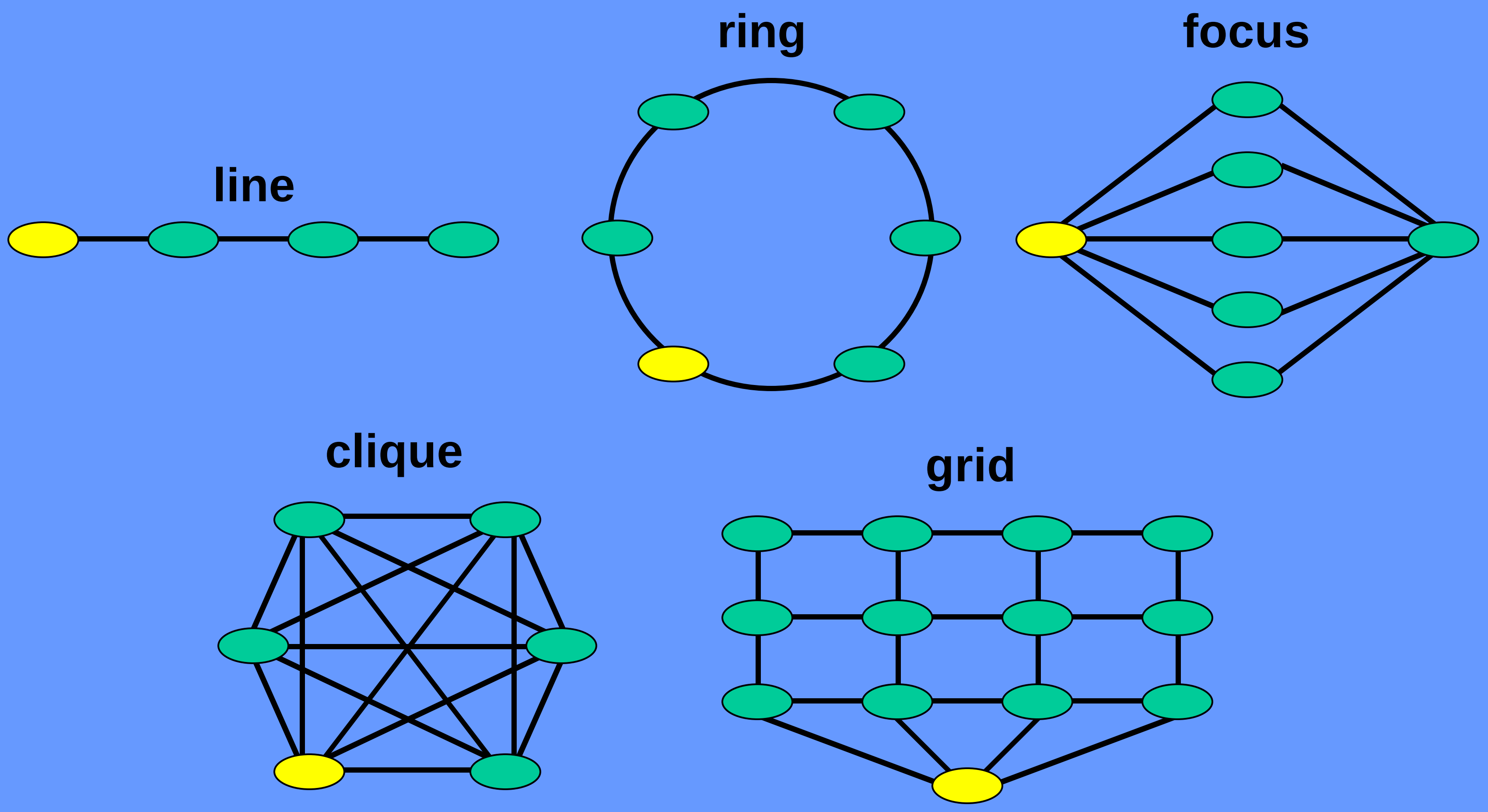
    [Labovitz et al, 1997-2000]

Goals

- overall
  - better understand dynamic behavior of BGP
- how does rate limiting impact convergence?
- precise analytical model?
  - seems unlikely ...
  - so we use simulation
  - and start small

Experiments

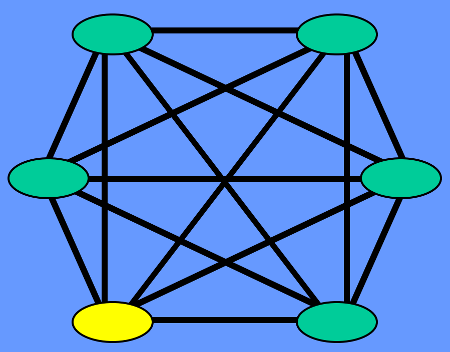
- simple topologies, simple policies



Experiments

- UP phase
  - advertise a single destination
- DOWN phase
  - withdraw a single destination

clique



Model Parameters

– size
– rate-limiting interval
– min & max processing times
– link delay
– sender-side loop detection
– withdrawal rate limiting
– jitter
– continuous rate-limiting
– random number seed index

Experiment DML

```
Net [ # the all-encompassing Net
  frequency 1000000000  # nanosecond simulation frequency
  randomstream [
    generator MersenneTwister
    stream 16512342004634582З
    reproducibility_level timeline
  ]

  Net [ id 1  AS_status boundary  router [ ... ] ]
  Net [ id 2  AS_status boundary  router [ ... ] ]
  ...

  link [ attach 1:1(1) attach 2:1(7) delay 0.01 ]
  link [ attach 1:1(2) attach 3:1(7) delay 0.01 ]
  ...

  bgpoptions [ ... ] # define global BGP options

] # end of the all-encompassing Net
```

Experiment DML

```
router [
 id 1
 graph [
  ProtocolSession [ name test use SSF.OS.BGP4.Widgets.Advertiser
                    workload_file /home/bj/blah  start_time 50 ]
  ProtocolSession [ name bgp  use SSF.OS.BGP4.BGPSession
                    autoconfig true ]
  ProtocolSession [ name socket use SSF.OS.Socket.socketMaster ]
  ProtocolSession [ name tcp  use SSF.OS.TCP.tcpSessionMaster ]
  ProtocolSession [ name ip  use SSF.OS.IP ]
  ProtocolSession [ name probe  use SSF.OS.ProbeSession
                    file "out.data"  stream "bgpstream" ]
 ]
]
```

Experiment DML

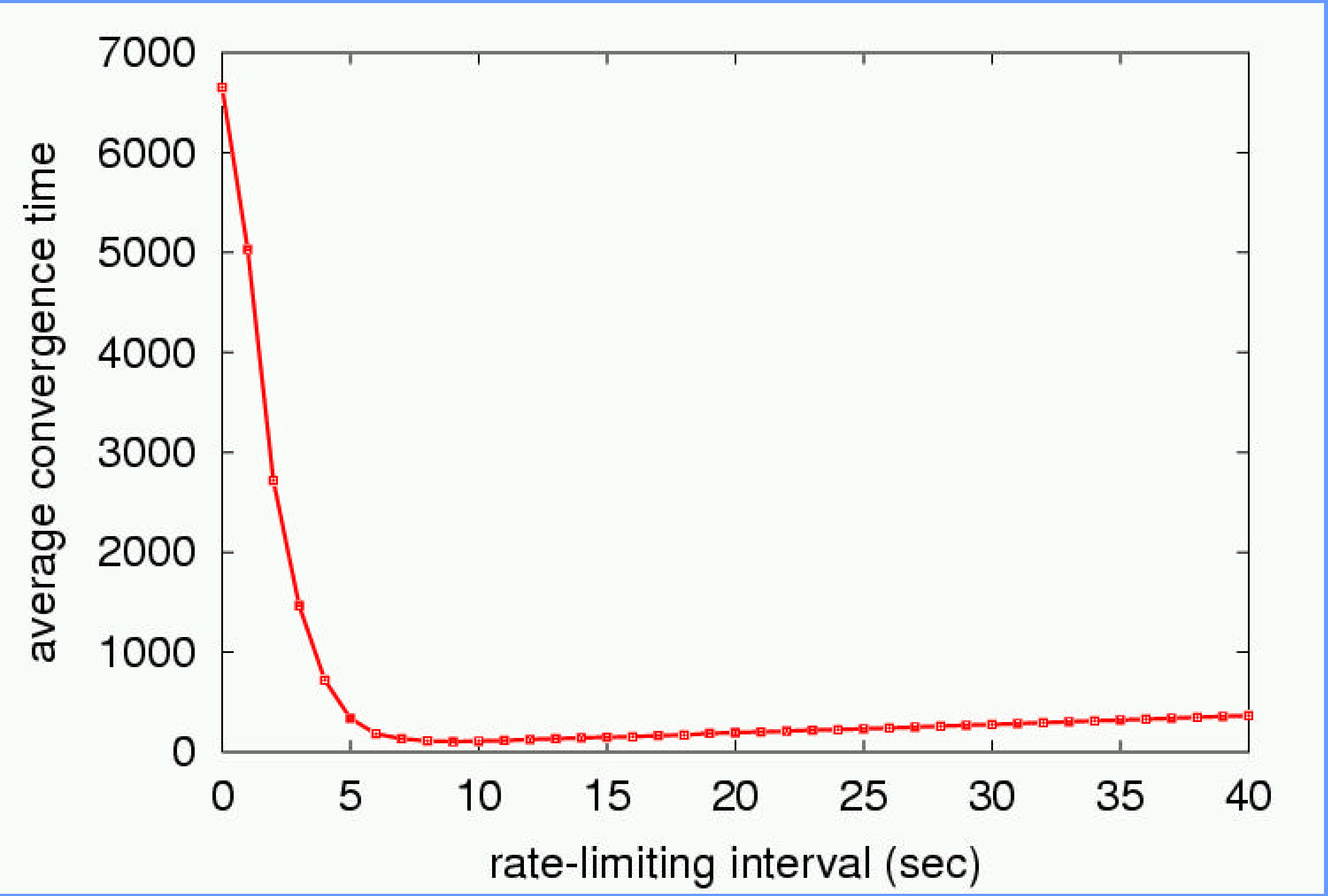
```
ProtocolSession [
  name bgp  use SSF.OS.BGP4.BGPSession  autoconfig false
  connretry_time 120
  min_as_orig_time 15
  reflector false
  neighbor [
    as 2  address 1(7)
    use_return_address 1(1)
    hold_time 90
    keep_alive_time 30
    mrai 10
    infilter [ _extends .filters.permit_all ]
    outfilter [ _extends .filters.permit_all ]
  ]
  neighbor [
    as 3  address 1(1)
    ...
  ]
]
```

Experiment DML

```
bgpoptions [ # define global BGP options
  show_conn_estab     true   # show connection establishment
  show_snd_update     true   # show when updates are sent
  ssld                false  # no sender-side loop detection
  auto_advertise      false
  show_fwd_table_add  true
  show_rcv_notif      true
  show_socket_events  false
  show_state_changes  false
  global_ebgp_mrai    20
  startup_jitter_bound 0.1
  # about 50 more
  ...
]
```

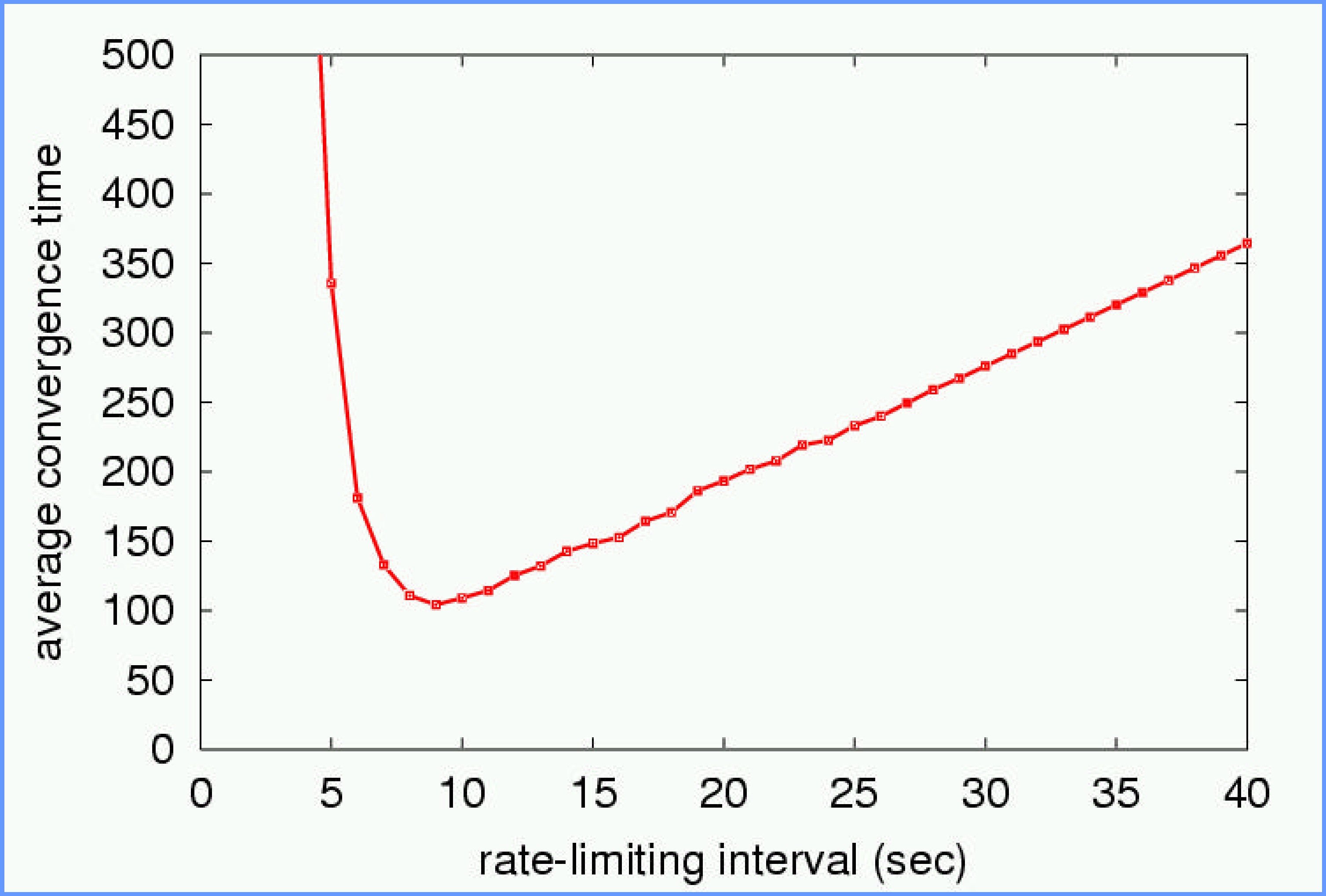
Average Convergence Time

clique size 15



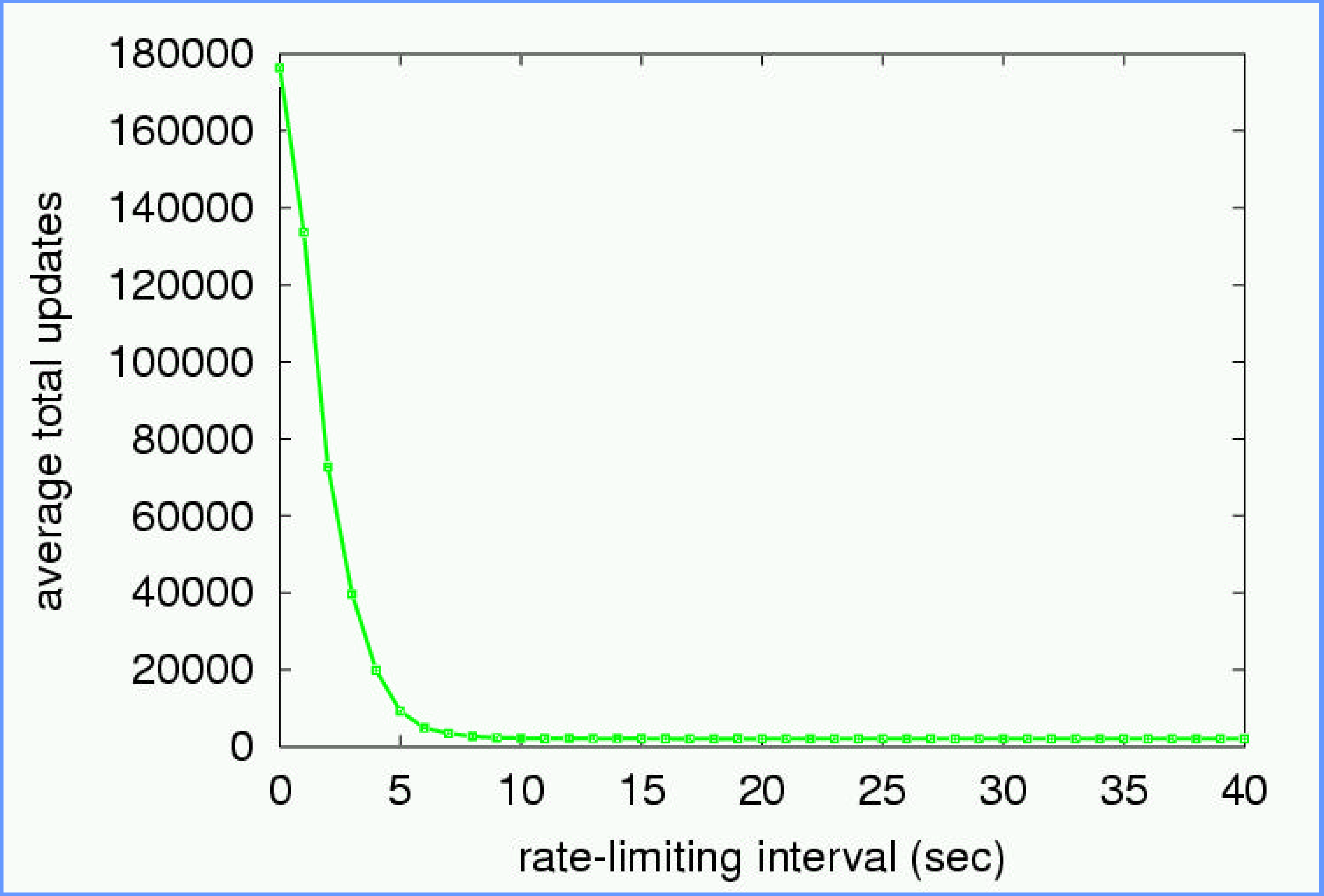
Average Convergence Time

clique size 15



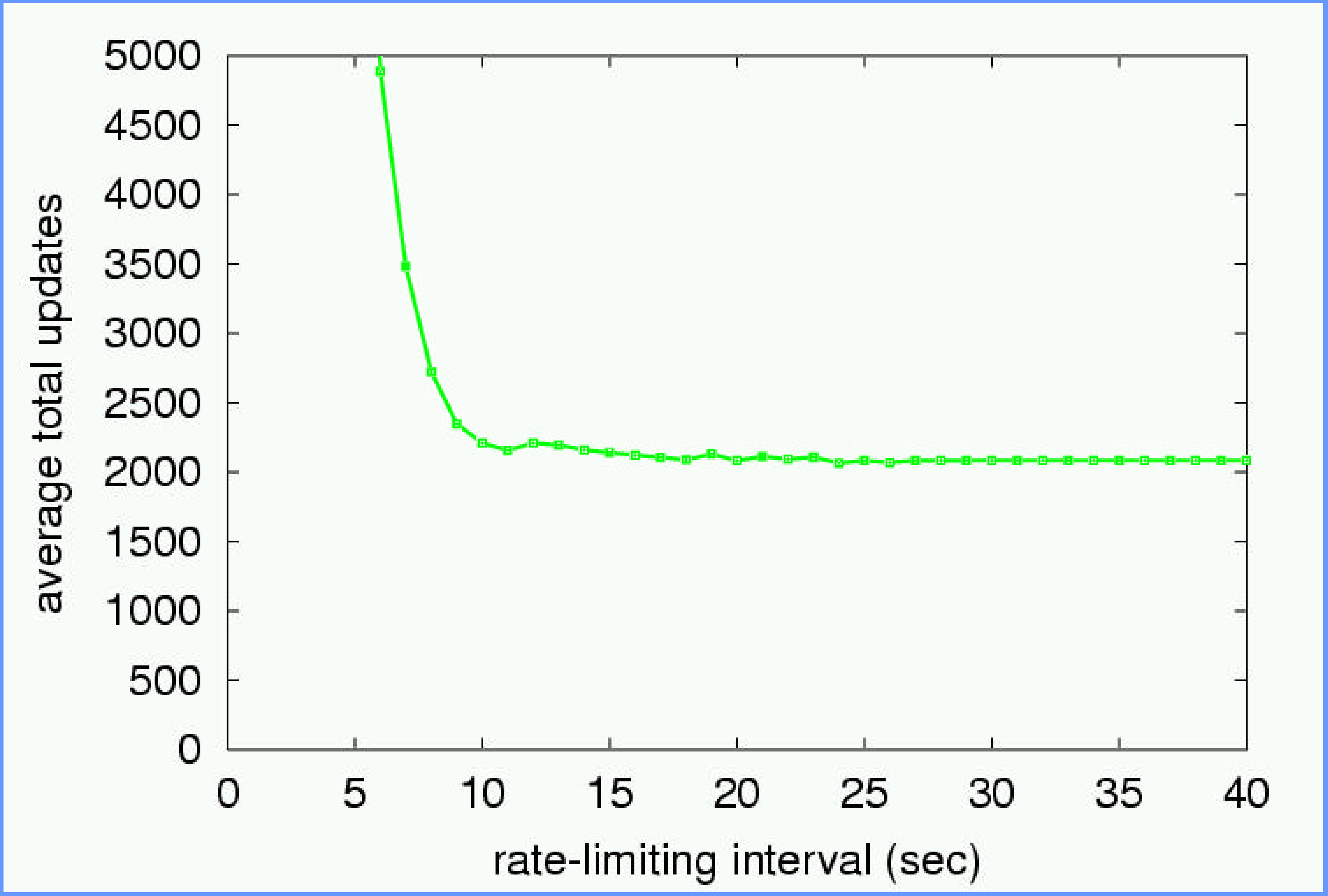
Average Total Updates

clique size 15



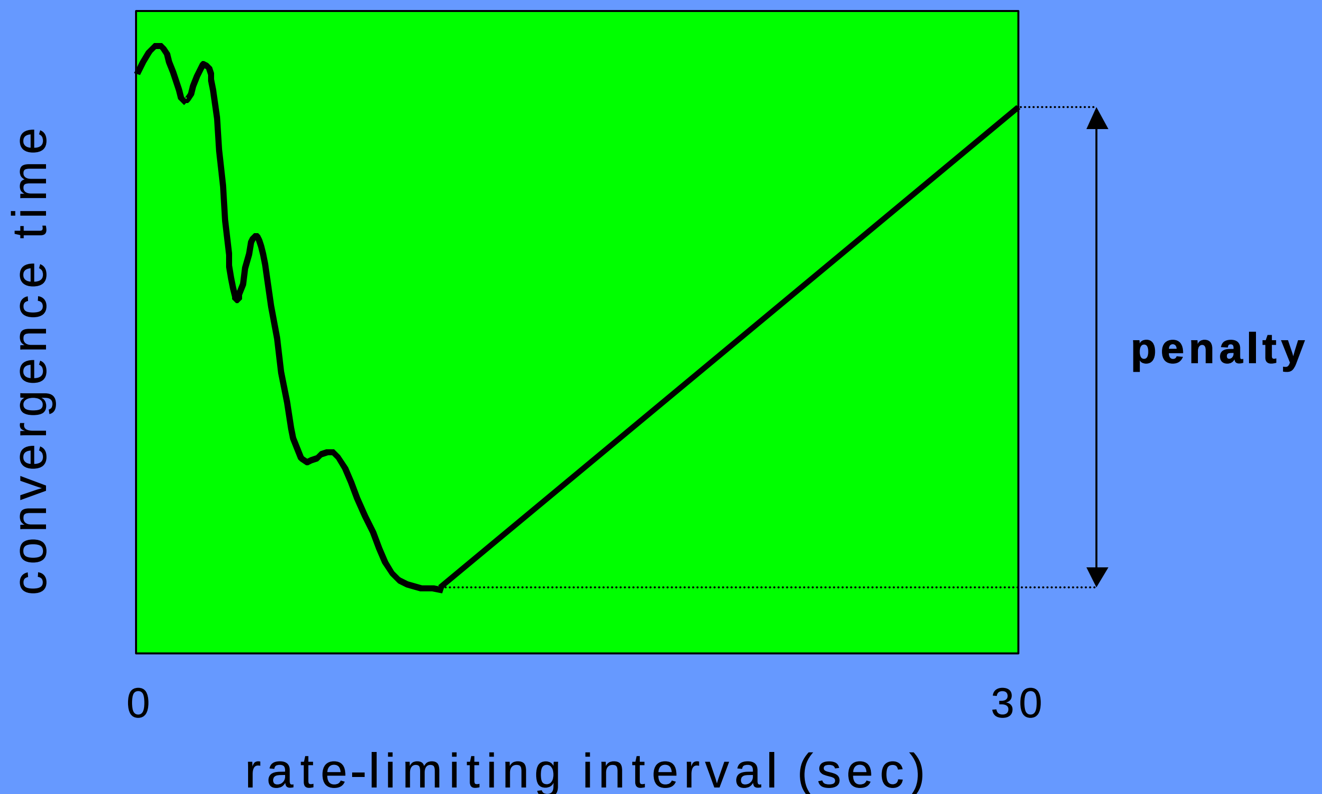
Average Total Updates

clique size 15



Generalized Results for Convergence Time

Observed optimal values much lower than values used in practice!



Continuing Work

- more realistic topologies and policies
- route flap dampening
    - long-term oscillations
- internal AS topologies
- multiple destinations
- per-route vs. per-peer MRAI
- accurate processing time models

SOS

- Scripts for Organizing Simulations
- Create families and groups of experiments
- Specify DML template, parameter values, and extractors
- Automatically generates DML, runs sets of experiments, extracts desired measurements
- Stores results in database

Documentation References

SSFNet & DML

(info & tutorials)

http://www.ssfnet.org/

DML

(tutorial)

http://www.cs.dartmouth.edu/~beej/talks/

SSFNet BGP

http://www.cs.dartmouth.edu/~beej/bgp/