Looking for Good Abstractions

timothy.griffin@cl.cam.ac.uk

INFOCOM 2006 Infrastructure Routing Panel

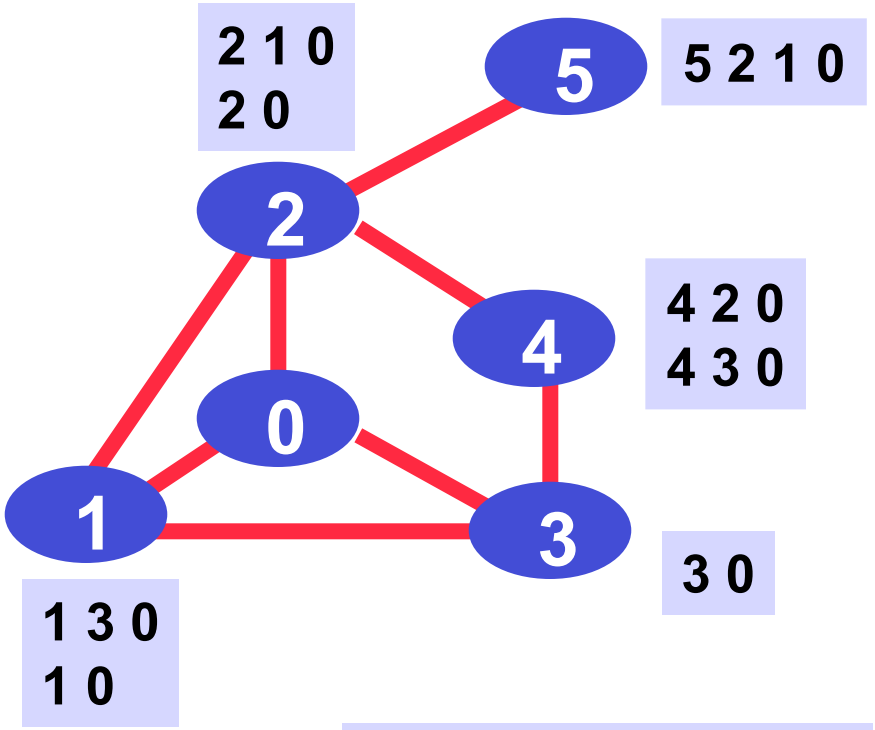
April 2006

What Problem is BGP Solving?

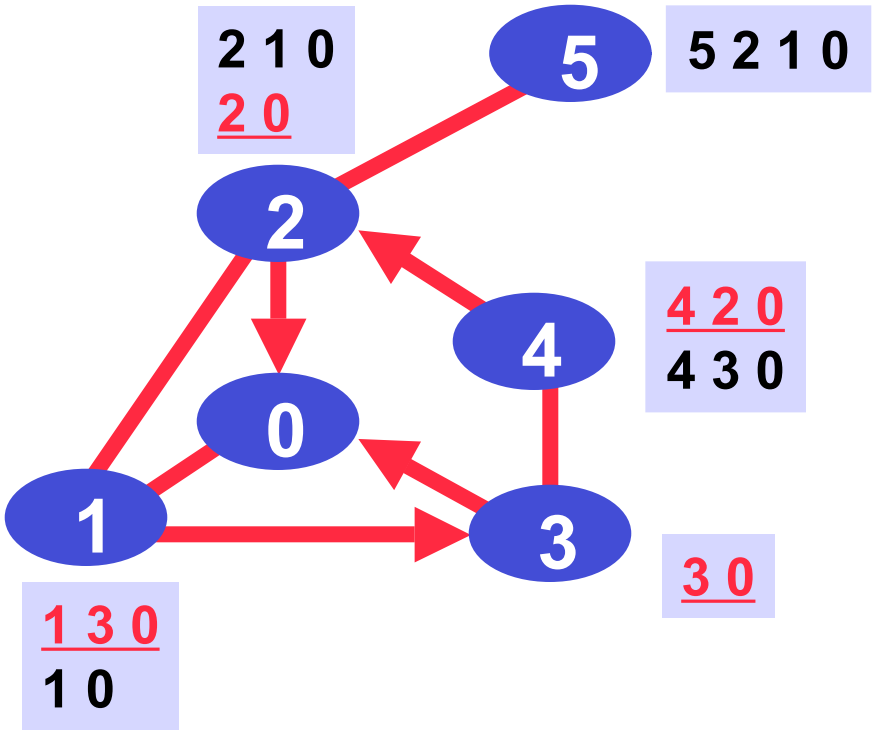
Underlying problem	Distributed means of computing a solution.
Shortest Paths	RIP, OSPF, IS-IS
Stable Paths	BGP

[GSW1998, GSW2002]

An instance of the *Stable Paths Problem*



A Solution to a Stable Paths Problem



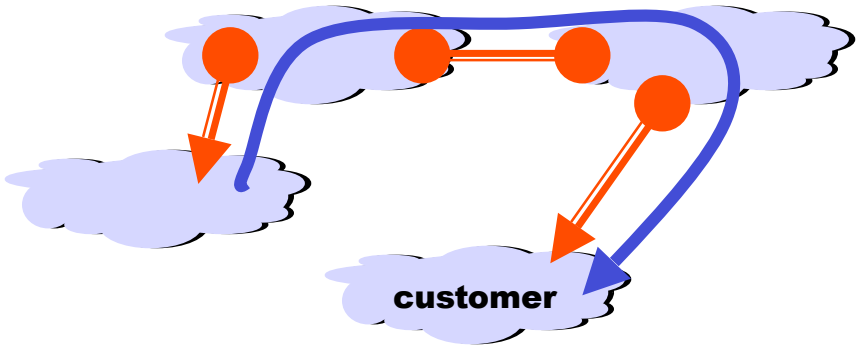
A sufficient condition for sanity

If an instance of SPP has an acyclic dispute digraph, then

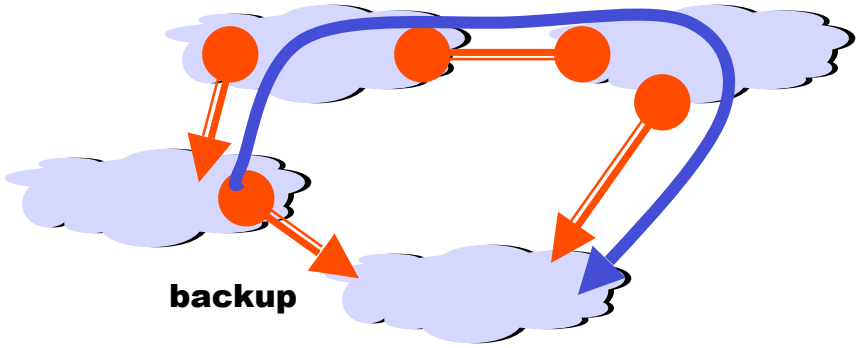
Static (SPP)	Dynamic (path-
solvable	safe (can't diverge)
unique solution	predictable restoration
all sub-problems uniquely solvable	robust with respect to link/node failures

RFC 4264 : BGP Wedgies

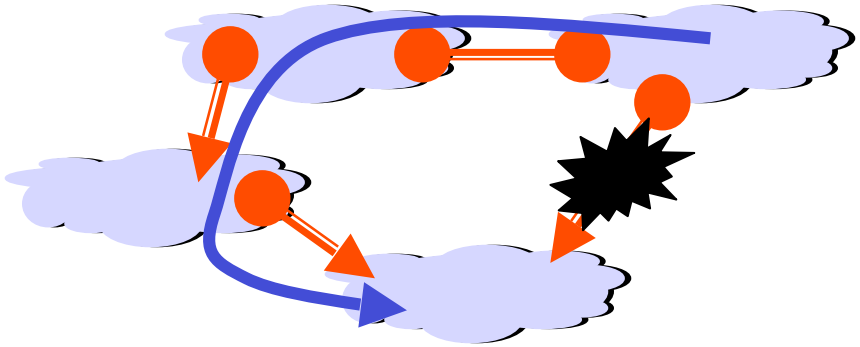
(Griffin, Houston)



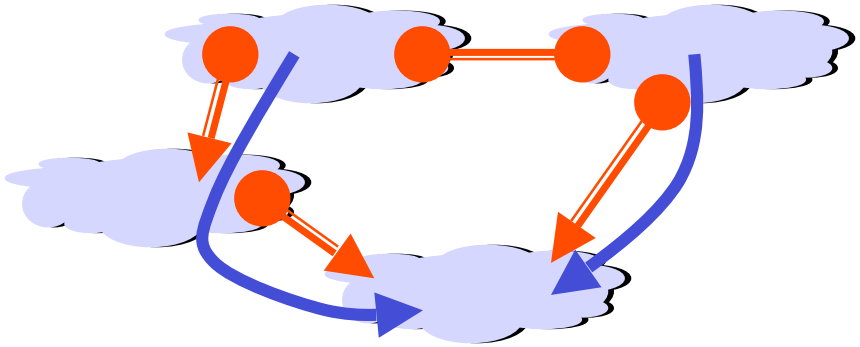
1



2 Install backup link using community



3 Disaster strikes primary link and the backup takes over



4 Primary link is restored but some traffic remains *pinned* to backup

Routing Algebras
João Luís Sobrinho

$$A = (\Sigma, \oplus, \otimes, \overline{0}, \overline{1})$$

Path Algebras --- 1970's, 1980s Gondran, Minoux, Carre', ...

$$A = (\Sigma, \leq, \otimes)$$

2002: Algebra and Algorithms for QoS Path Computation and Hop-by-Hop Routing in the Internet.

$$A = (\Sigma, \leq, \Lambda, \otimes)$$

2003: Network Routing and Path Vector Protocols: Theory and Applications. (SIGCOMM)
2005: An Algebraic Theory of Dynamic Network Routing (TON)

Metarouting

(Griffin, Sobrinho SIGCOMM 2005)

Can we do for routing protocols what YACC did for parsers?

BGP on one page

(from a current prototype implementation)

```
let prefix : algebra =
   op(isolate(IPv4))

let lp3 : algebra =
   lp(min(0,3))

let cpp : algebra =
   fm(lp3)

let node_path : algebra =
   slists(100, strings(20))

let community_set : algebra =
   tags(100, 20)

let sp : algebra =
   add(1, 1000)
```

```
let ebgp : algebra =
    lex <
         nlri : prefix,
          loc : cpp,
         path : node_path,
         comm : community_set,
            d : lp(sp),
        ipath : lp(node_path),
        icomm : lp(community_set)
        >

let ibgp : algebra =
    lex <
         nlri : prefix,
          loc : op(cpp),
         path : op(node_path),
         comm : op(community_set),
            d : sp,
        ipath : node_path,
        icomm : community_set
        >

let bgp : algebra = lunion <ebgp : ebgp, ibgp : ibgp>
```

Familiar things through fresh eyes

DEFAULT ADMINISTRATIVE DISTANCE

direct interface	0
static route	1
EIGRP Summary Route	5
External BGP	20
IGRP	100
OSPF	110
IS-IS	115
RIP	120
I-BGP	200

$$A \equiv \vec{\Pi} i \in \{0,1,\cdots,n\} : A_i$$

Open Problems

- Good metarouting Language design
- Addressing and Forwarding?
- Tunnels as first-class objects
- 2547-ish VPNs?