NOKIA

Bell Labs

SDN Enabled Path Switching

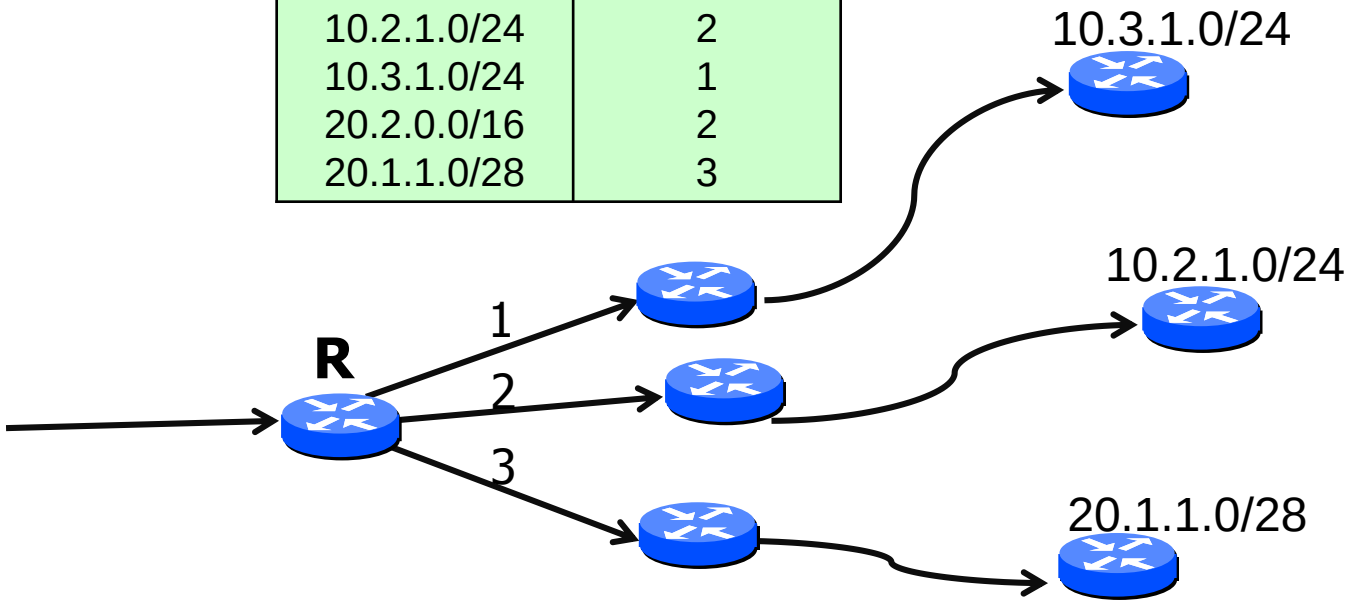
- Gordon Wilfong Bell Labs
- Joint work with Adiseshu Hari (Bell Labs) and Urs Niesen (Qualcomm)

How is routing currently done?

Destination-Based Forwarding Table

R:

Destination	Output Port
10.1.0.0/24	3
10.1.2.0/24	2
10.2.1.0/24	2
10.3.1.0/24	1
20.2.0.0/16	2
20.1.1.0/28	3



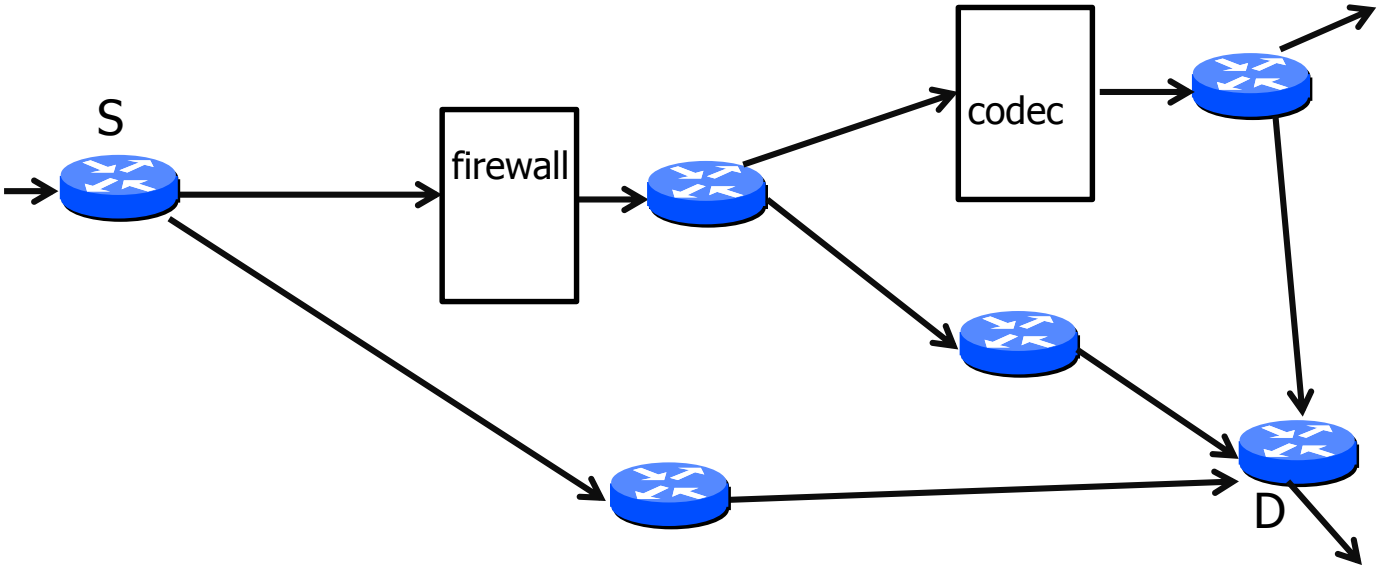
Destination-based Routing Advantages

- Destination-based routing tables populated locally by distributed routing protocols (e.g., BGP or RIP or …)

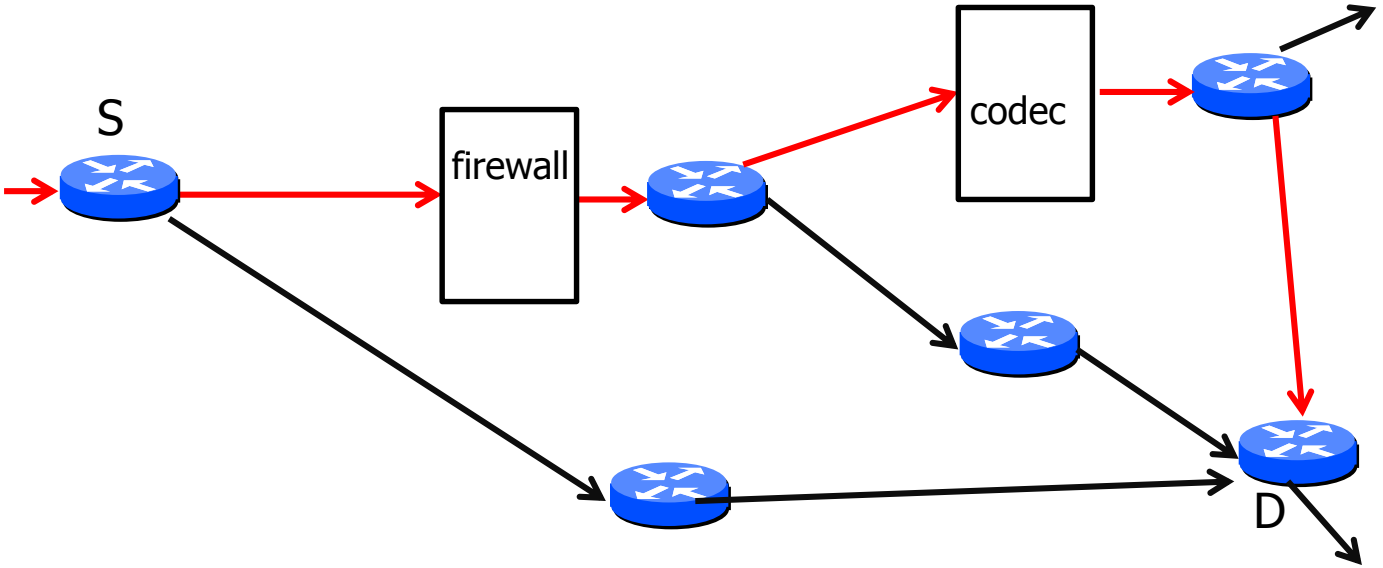
Destination-based Routing Disadvantages

- Destination-based tables grow as address blocks fragment and increase in number
- Does not allow for flexible policy-based routing

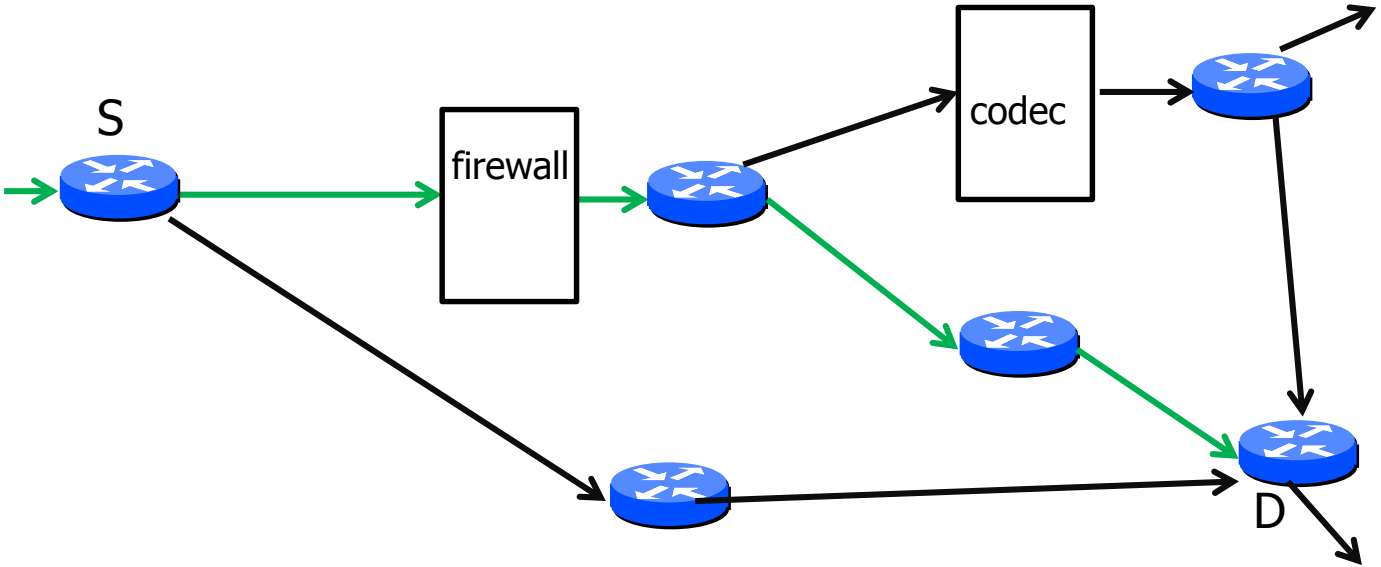
Policy-Based Routing



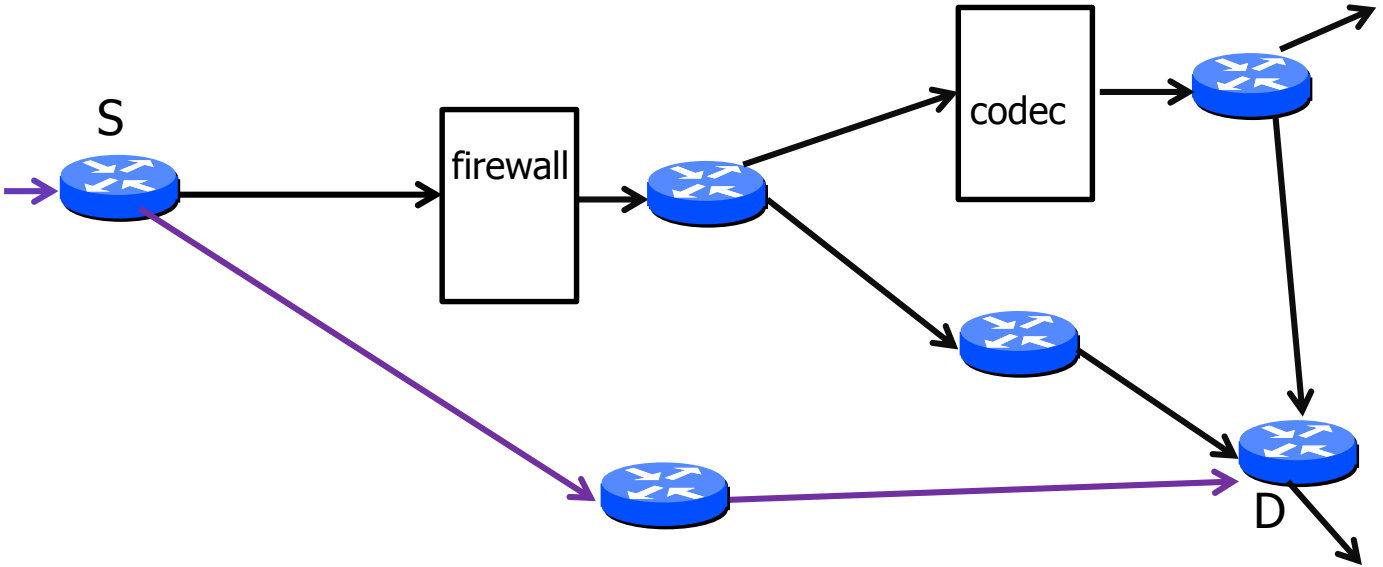
Policy-Based Routing



Policy-Based Routing



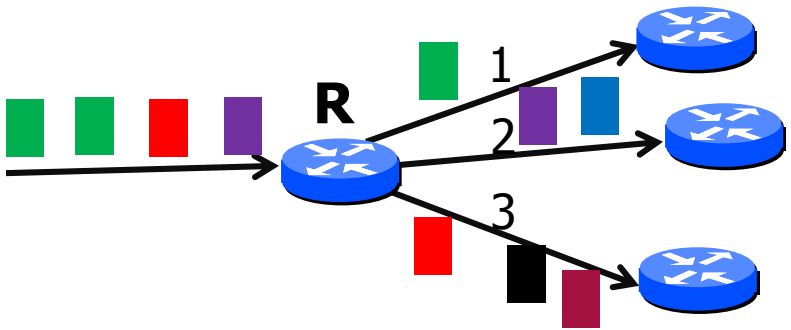
Policy-Based Routing



Flow-Based Forwarding Table

R:

Flow	Output Port
Red	3
Blue	2
Purple	2
Green	1
Black	3
??	3



Flow-based Routing Advantages

- Allows for flexible policy-based routing

Flow-based Routing Disadvantages

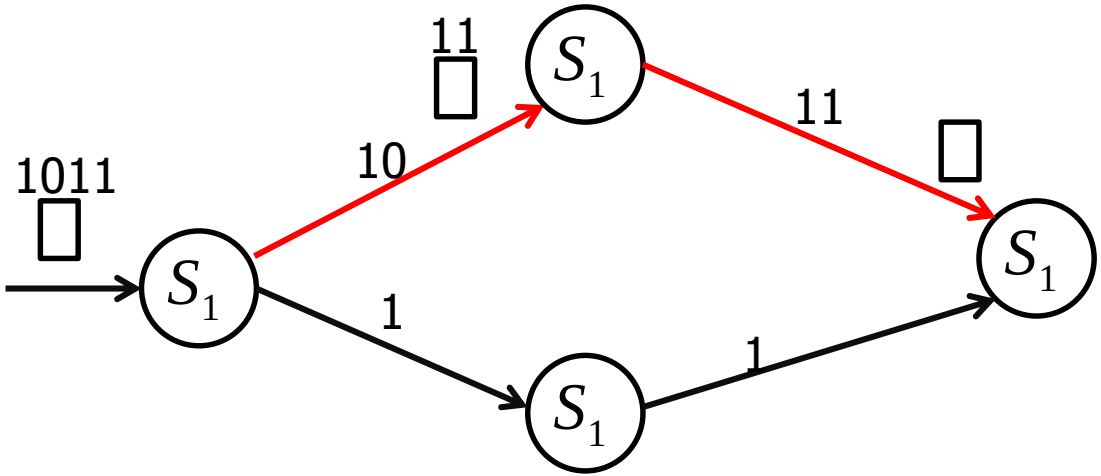
- Flow-based tables grow as more and more complex policies are introduced

Q: Is there a way to allow for unbounded growth in policy-complexity (i.e., growth in number of flows) while keeping forwarding table size fixed?

Our proposal

Path Encoding

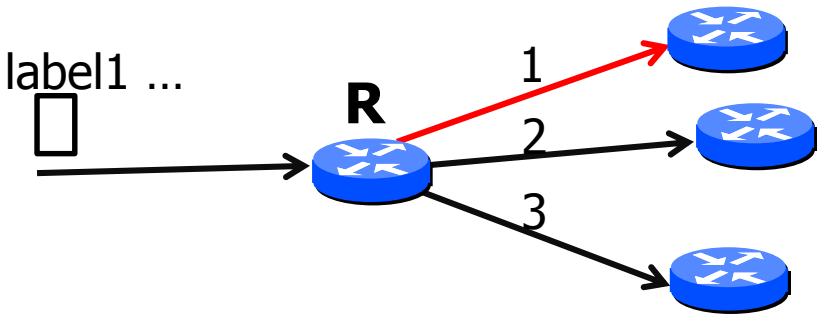
- For each node, the outgoing ports are given a distinct binary label
- Each packet header includes path encoded as a sequence of port labels



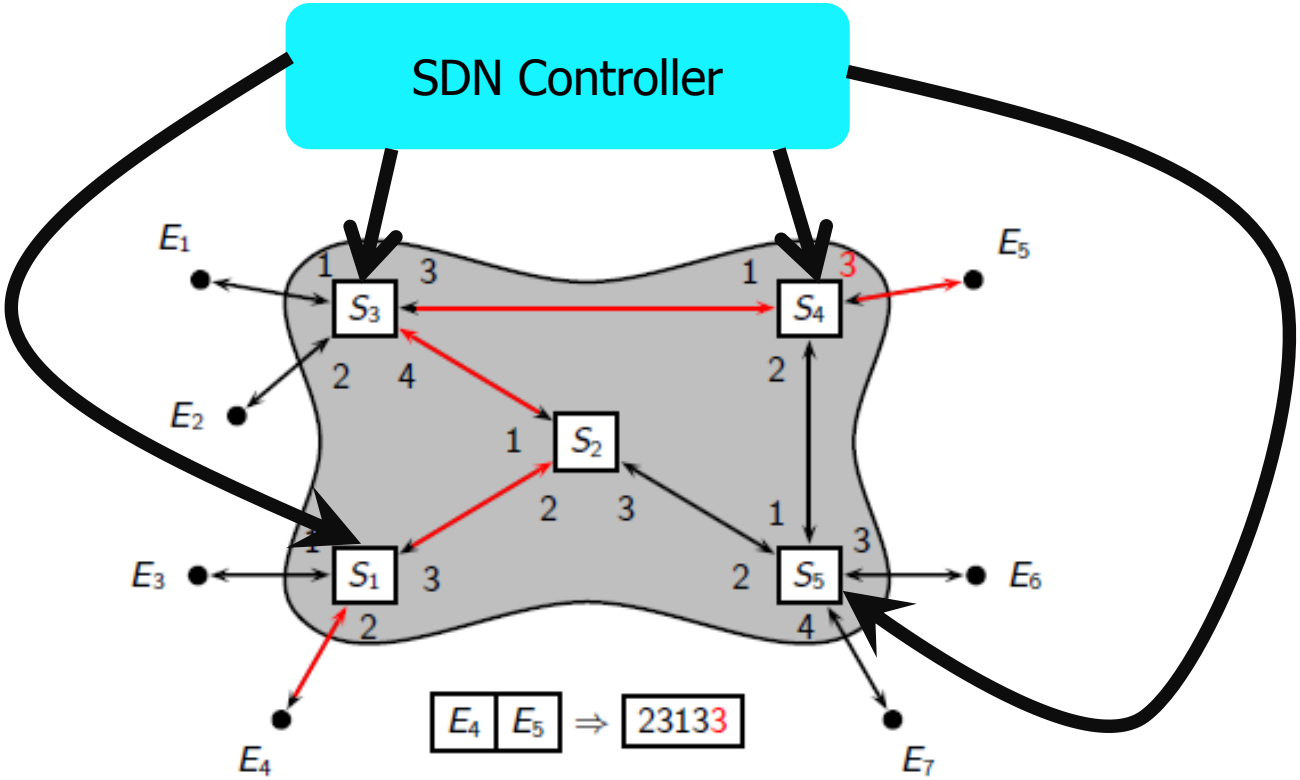
Path Encoding Table

R:

Label	Output Port
label1	1
label2	2
label3	3



Software defined network



Path Encoding Advantages

- Table size is bounded by the number of ports
- As flows or destinations increase, table size remains fixed

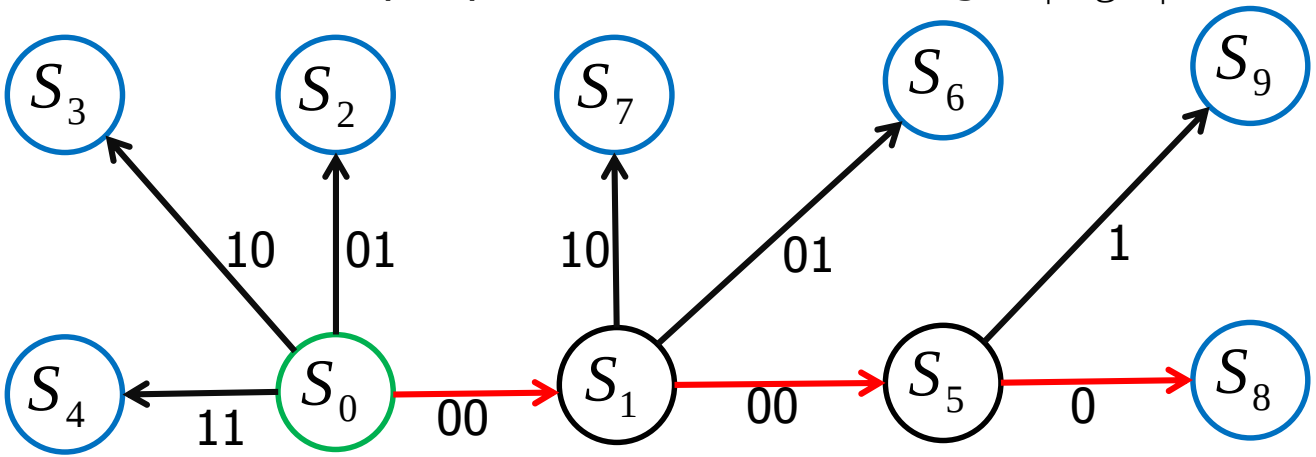
Path Encoding Disadvantages

- Takes up space in packet headers

Fixed-length Labels

In the next few slides, we consider paths from S_0 to all leaf nodes.

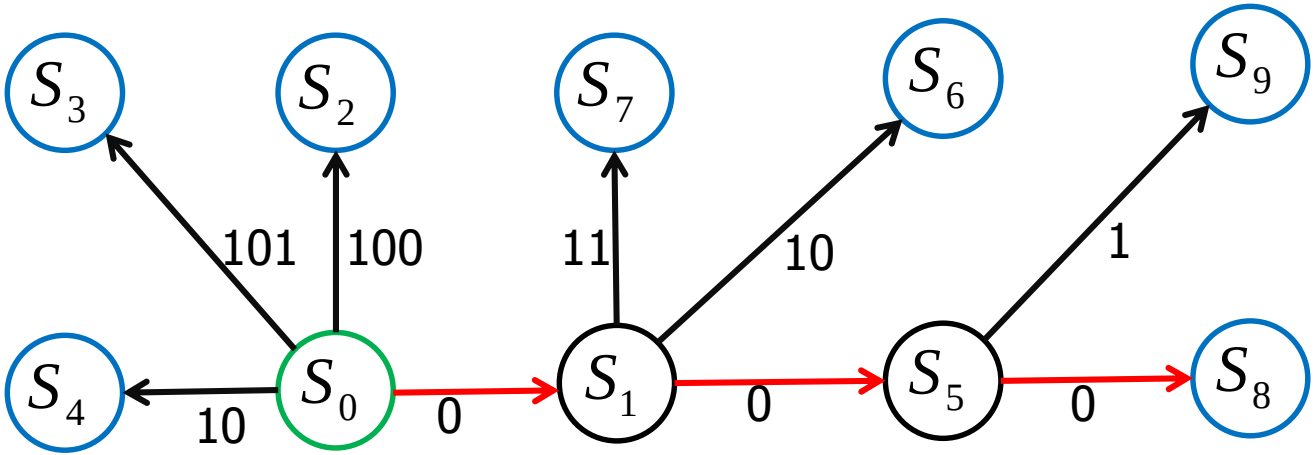
A node with k output ports uses labels of length $\lceil \log k \rceil$



Longest path encoding from S_0 to a leaf is 5,

e.g., encoding 00000 of path (S_0, S_1, S_5, S_8)

Variable-length Binary Labels

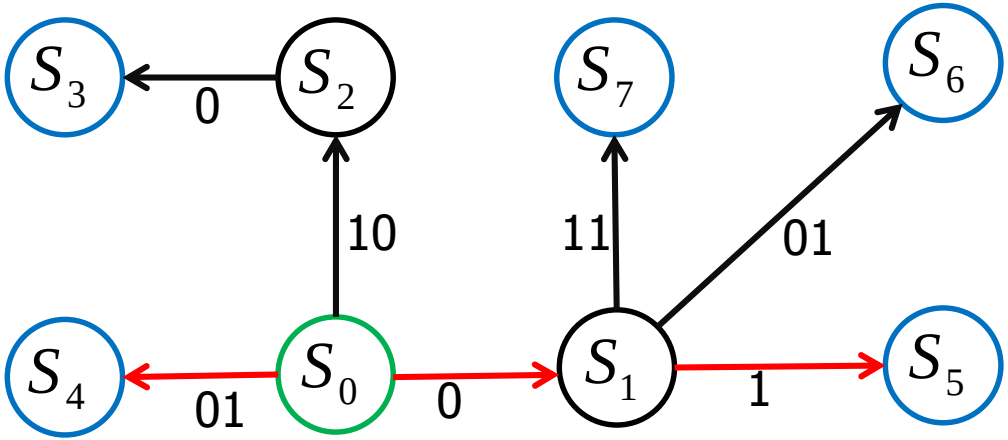


Longest path encoding from S_0 to a leaf has length 3

e.g., encoding 000 of path (S_0, S_1, S_5, S_8).

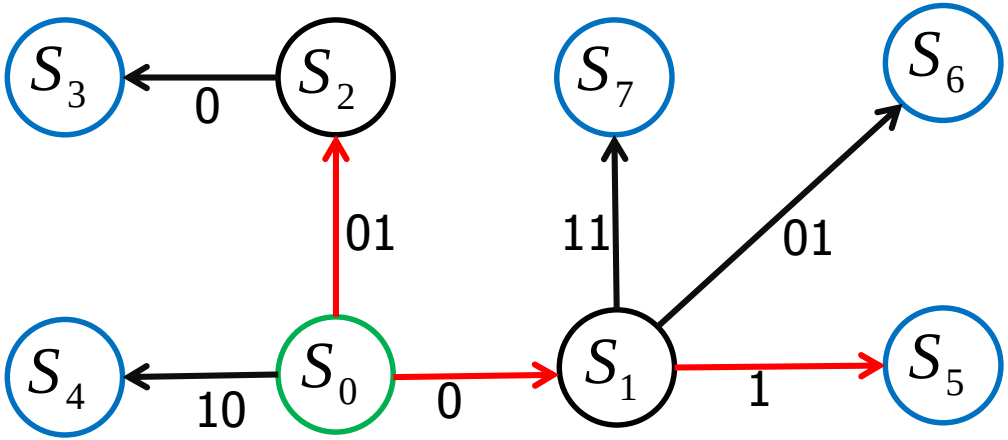
How to unambiguously decode a path label?

Decoding Variable-length Binary Labels: Uniqueness



- Path labels may not be unique
  - (S_0, S_1, S_5) and (S_0, S_4) both have label 01

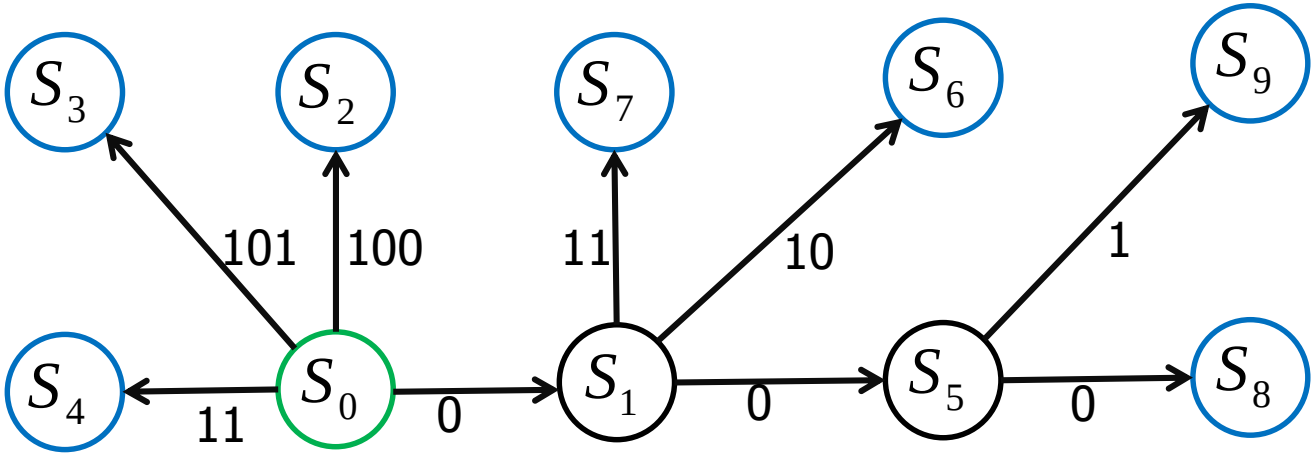
Decoding Variable-length Binary Labels: Local



- Path labels may not be locally decodable even if encodings are unique
  - S_0 does not know that sending packet out port labeled 01 will not end at leaf
  - (S_0, S_1, S_5) is unique path to leaf with label 01 and so S_0 should send packets labeled 01 on port labeled 0
  - Nodes also should not have to know labels at other nodes to decode

Prefix-free Labeling (PFL)

PFL: At each node, no label is a prefix of another label at that node.



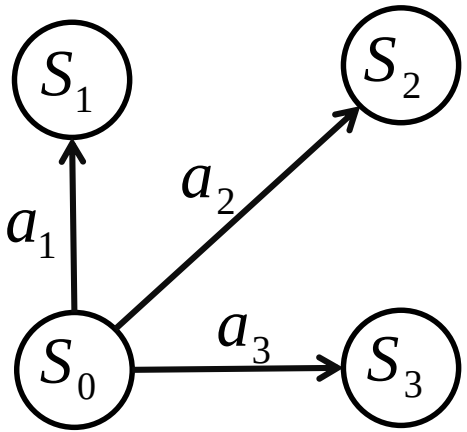
- PFL implies distinct path encodings for paths from same source
- PFL implies unambiguous decoding at each node with only local information

Computing a prefix-free labeling

Preliminaries

Directed graph: $D = (V, A)$

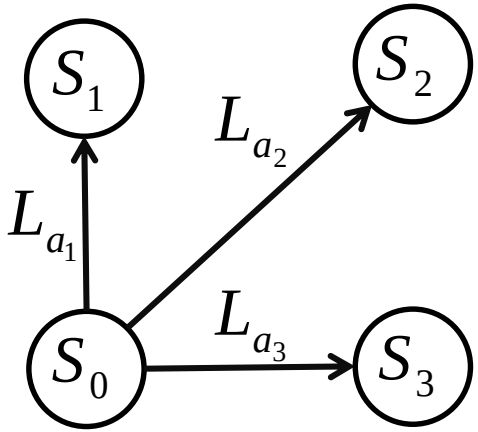
Def: $out(S) = \{a \in A : tail(a) = S\},\ S \in V$



$out(S_0) = \{a_1, a_2, a_3\}$

Arc Label Lengths

Def: Length of binary label of arc a = L_a



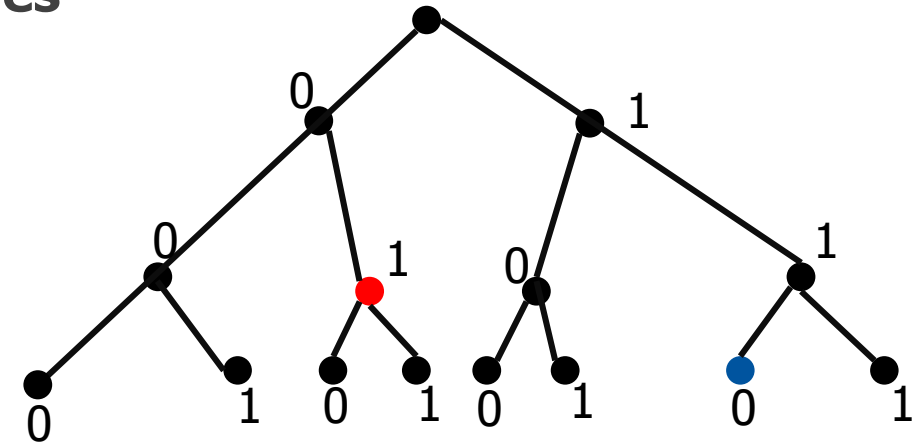
Kraft's Inequality

A code with codewords $w_1, w_2, \ldots, w_k$ having

lengths $L_1, L_2, \ldots, L_k$ is said to satisfy Kraft's Inequality if:

$$\sum_{1 \le i \le k} \frac{1}{2^{L_i}} \le 1$$

Tree Codes



● = 01

● = 110

Prefix-free if no two chosen nodes on the same path to the root

Prefix-free Code ⇔Kraft's Inequality

Theorem: A prefix-free code with codewords having lengths

$\{L_1, L_2, \ldots, L_k\}$ exists if and only if the lengths satisfy Kraft's Inequality.

Proof: Assume wlog $L_1 \leq L_2 \leq \ldots \leq L_k$.

Suppose lengths satisfy Kraft's Inequality.

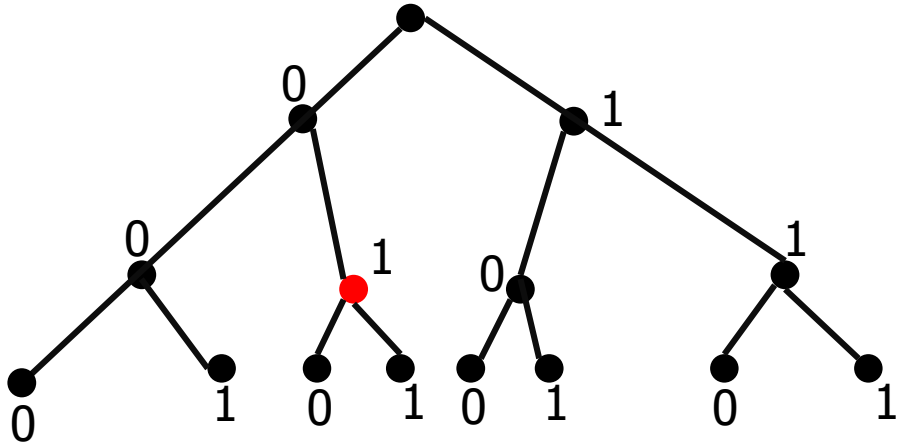
Start with full binary tree of depth L_k .

At step i: choose a node at depth L_i to determine codeword and remove all descendants.

Must show this is always possible for steps i: $1 \leq i \leq k$

Example

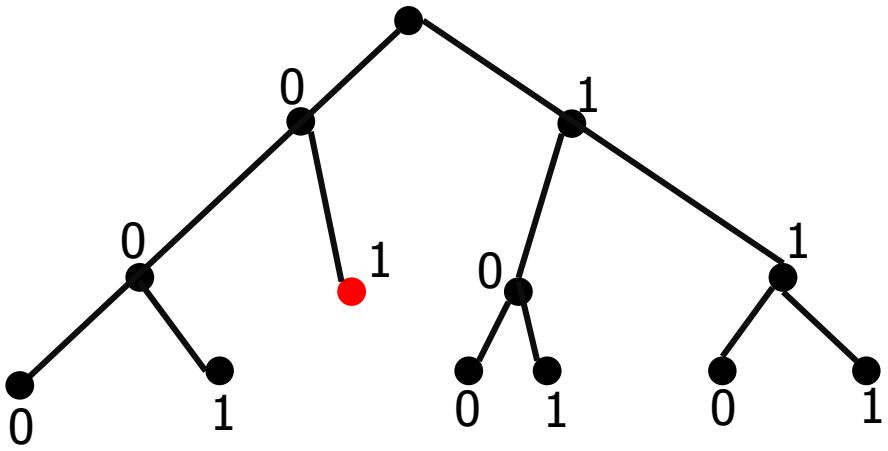
$L_i = 2$



● = 01

Example

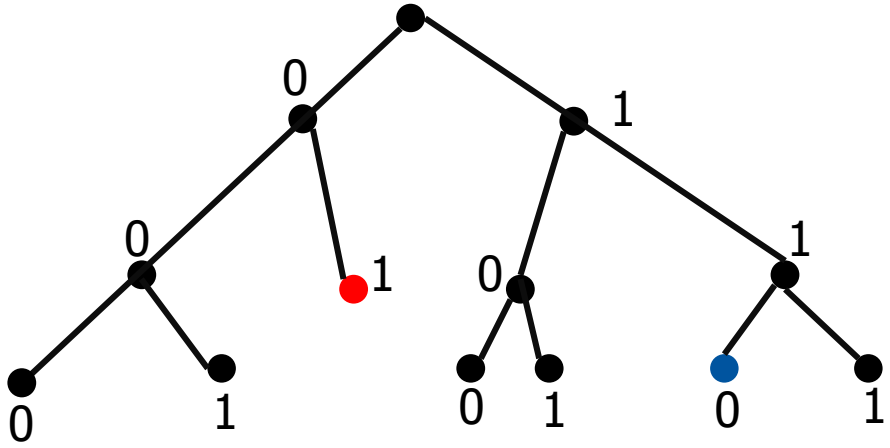
Remove descendants of ●



● = 01

Example

$L_{i+1} = 3$



● = 01

● = 110

Proof continued

We show that at step i there is at least one remaining leaf at depth L_i .

Number of remaining leaves at step i is:

$$2^{L_k} - \sum_{j=1}^{i-1} 2^{L_k - L_j} = 2^{L^k}\left(1 - \sum_{j=1}^{i-1} 2^{-L_j}\right) > 2^{L_k}\left(1 - \sum_{j=1}^{k} 2^{-L_j}\right) \geq 0$$

Kraft's Inequality

Proof continued … again

- Suppose it is a prefix-free code.
- Consider a complete binary tree of depth L_k .
- A length L_i codeword has a set S_i of leaves under it where $|S_i| = 2^{L_k - L_i}$

and $S_i \cap S_j = \emptyset$ for $i \neq j$.

$$\sum_{i=1}^{k} |S_i| = \sum_{i=1}^{k} 2^{L_k - L_i} \leq 2^{L_k} \Rightarrow \sum_{i=1}^{k} 2^{-L_i} \leq 1$$

total number of leaves

Kraft's Inequality

■

Definitions

Definition: A valid labeling is a labeling of the arcs in A where if L_a is the length of the label of arc a then for each $v \in V$ the label lengths of outgoing arcs

satisfy Kraft's Inequality $\sum_{a \in out(v)} 2^{-L_a} \leq 1$

Definition: The length of a path P is $L(P) = \sum_{a \in P} L_a$

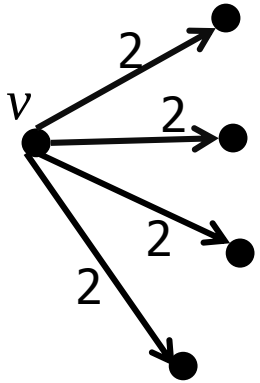
Problem statement

Given a digraph D=(V,A) and a set of paths $\mathcal{P} = \{P_i : 1 \le i \le k\}$ in D.

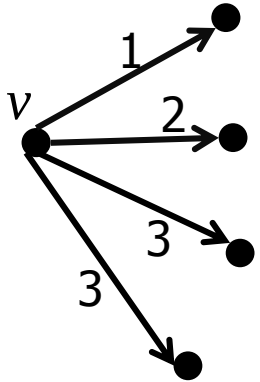
Find a valid labeling of the arcs of A to minimize $\max\{L(P_i) : 1 \le i \le k\}$.

We call this the **optimal path encoding problem.**

Examples



$$\frac{1}{4}+\frac{1}{4}+\frac{1}{4}+\frac{1}{4}=1$$



$$\frac{1}{2}+\frac{1}{4}+\frac{1}{8}+\frac{1}{8}=1$$

Optimal Path Encoding is APX-hard

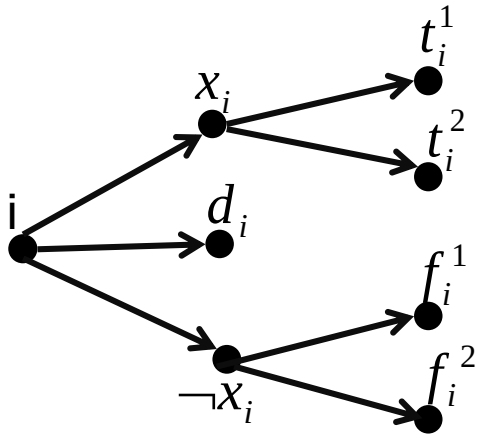
Theorem: Approximating the optimal path-encoding problem to within a factor better than 8/7 is NP-hard.

Proof: Reduction from SAT where each clause has 2 or 3 literals and each *variable* is involved in at most 3 clauses [Tovey84].
=> Each *literal* in 1 or 2 clauses

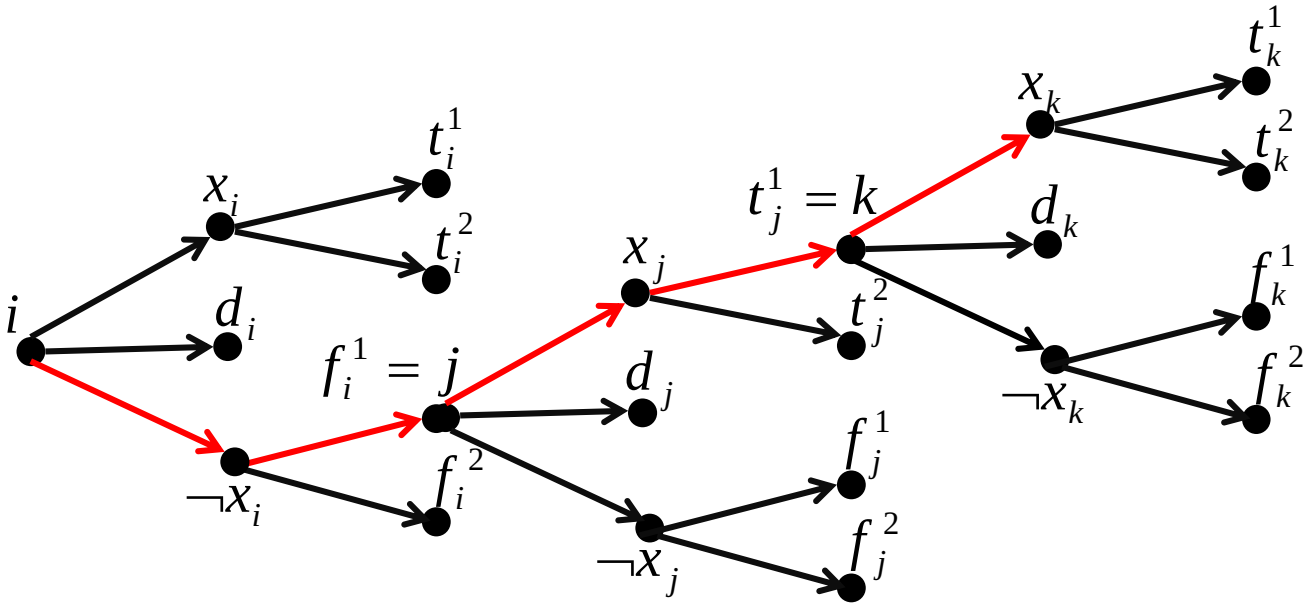
$$C_1 = (x_i, \_, \_)$$
$$C_2 = (\_, x_i, \_)$$
$$C_3 = (x_i, \_, \_)$$

$$C_1 = (x_i, \_, \_)$$
$$C_2 = (\_, \neg x_i, \_)$$
$$C_3 = (x_i, \_, \_) \text{ or}$$
$$C_3 = (\neg x_i, \_, \_)$$

Gadget for variable x_i

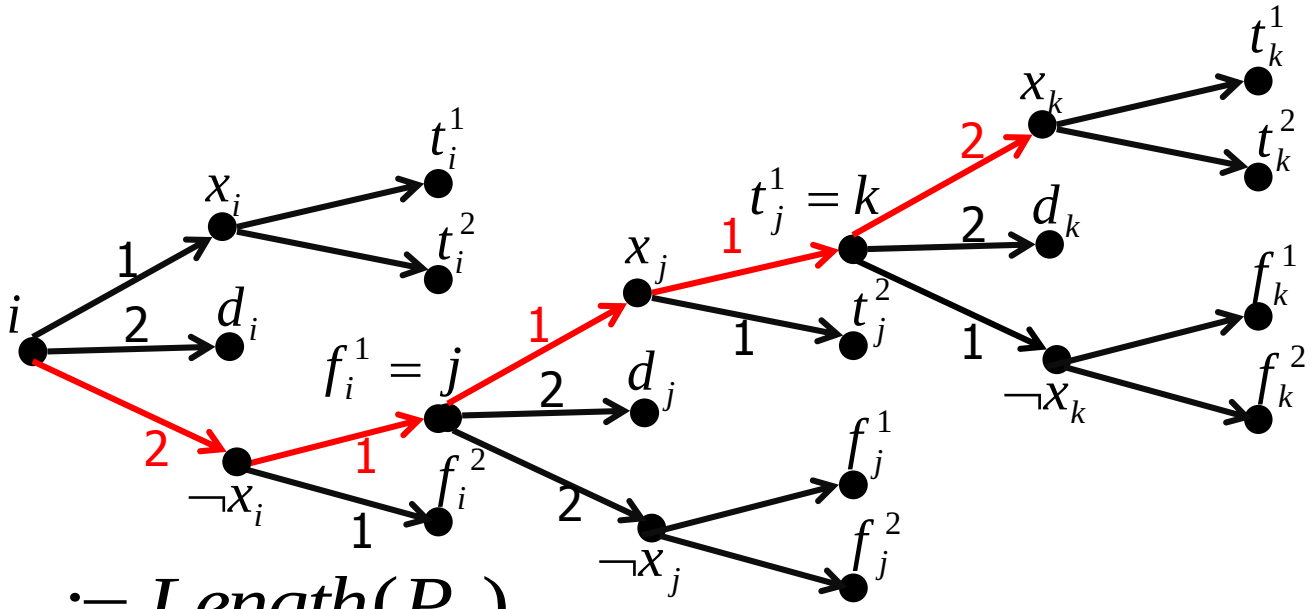


Gadget for clause $C = (\neg x_i \vee x_j \vee x_k)$



P_C

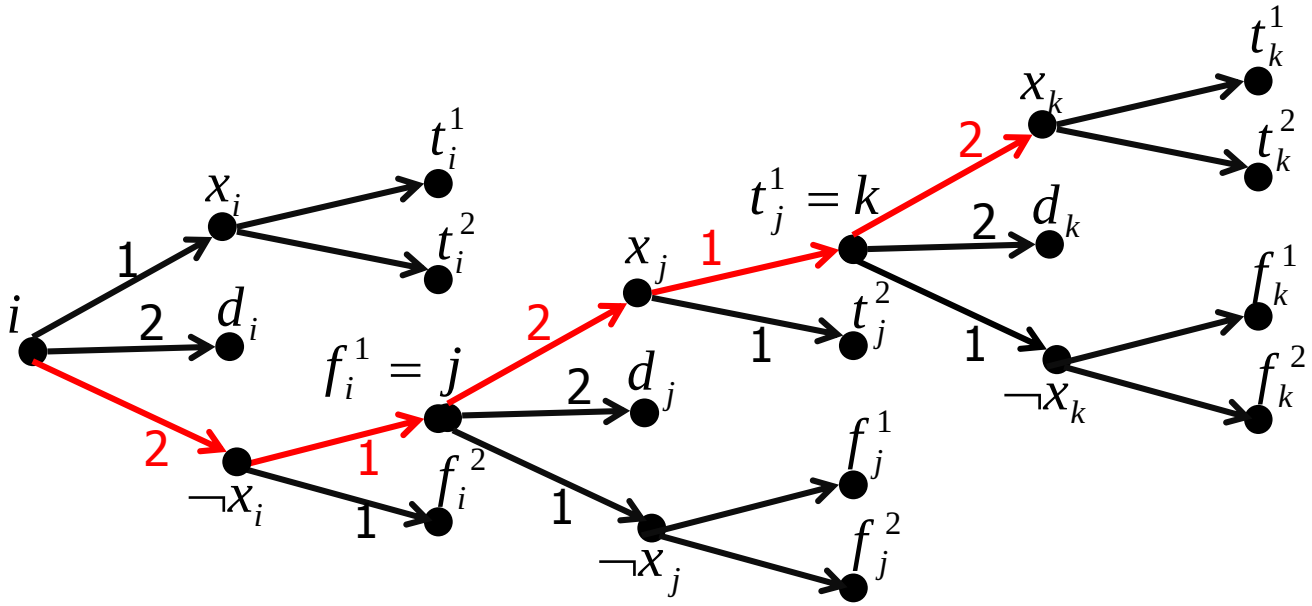
Gadget for clause $C = (\neg x_i \vee x_j \vee x_k)$



$$L_C := Length(P_C)$$

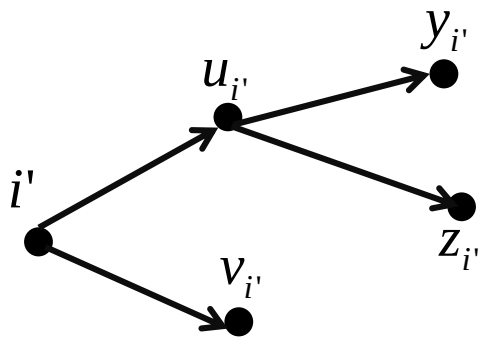
C satisfied by $x_i = true, x_j = true, x_k = false \Rightarrow L_C = 7$

Gadget for clause $C = (\neg x_i \vee x_j \vee x_k)$

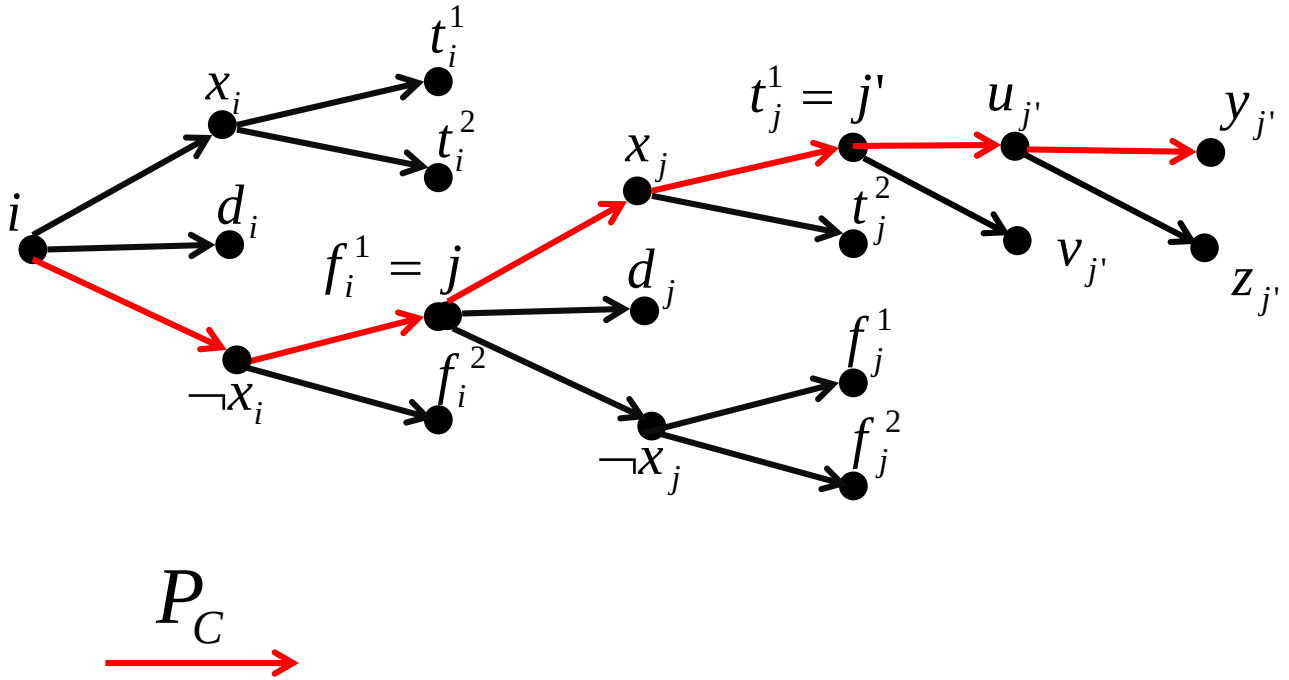


C not satisfied $x_i = true, x_j = false, x_k = false \Rightarrow L_C = 8$

Dummy gadget for variable X_i



Gadget for clause $C = (\neg x_i \vee x_j)$



Optimal Path Encoding: Convex Minimization Problem

$$\min L$$

$$\text{subject to}:$$

$$\sum_{a \in P_i} L_a \le L \ , \forall P_i \in \mathbf{P}$$

$$\sum_{a \in out(v)} 2^{-L_a} \le 1, \forall v \in V \quad \Longrightarrow \quad L_a \ge 0, \forall a \in A \qquad \text{(Kraft)}$$

$$L \ \in Z$$

$$L_a \in Z, \forall a \in A$$

Optimal solution: $\left(L^I, \left\{L_a^I\right\}\right)$

Relaxed version

$$\min L$$

$$\text{subject to:}$$

$$\sum_{a \in P_i} L_a \leq L \ , \forall P_i \in \mathcal{P}$$

$$\sum_{a \in out(v)} 2^{-L_a} \leq 1, \forall v \in V \quad \text{or} \quad \ln\left(\sum_{a \in out(v)} \exp(-\ln(2) L_a) \right) \leq 0$$

Geometric program
(efficiently solved by interior point methods)

$$L \in R$$

$$L_a \in R, \forall a \in A$$

Optimal solution: $\left(L^*, \{ L_a^* \} \right)$

Rounding Relaxed Solution

Define $L_a^R = \left\lceil L_a^* \right\rceil, \forall a \in A$

$$L^R = \max_{p \in P} \sum_{a \in p} L_a^R$$

Solutions

$\left(L^I, \{L_a^I\}\right)$ Optimal integer solution (what we want)

$\left(L^*, \{L_a^*\}\right)$ Optimal relaxed solution (what we can get)

$\left(L^R, \{L_a^R\}\right)$ Rounded optimal relaxed solution (settle for this)

Useful Fact

Lemma: If $L_a^I = 0$ wlog we can assume $L_a^* = 0$

Proof: $L_a^I = 0$ only if a is the only outgoing arc at tail(a). ■

Fact: $L_a^* > 0 \Rightarrow L_a^I > 0$

Rounding Gives a 2-Approx

Theorem: Rounded solution is valid and $L^I \le L^R \le 2L^I$

Proof: $\sum_{a \in out(v)} 2^{-L_a^R} \le \sum_{a \in out(v)} 2^{-L_a^*} \le 1$ (hence valid labeling)

Let p be a path where $L^R = \sum_{a \in p} L_a^R$

$$\le \sum_{a \in p} L_a^* + \left|\left\{a \in p : L_a^* > 0\right\}\right|$$

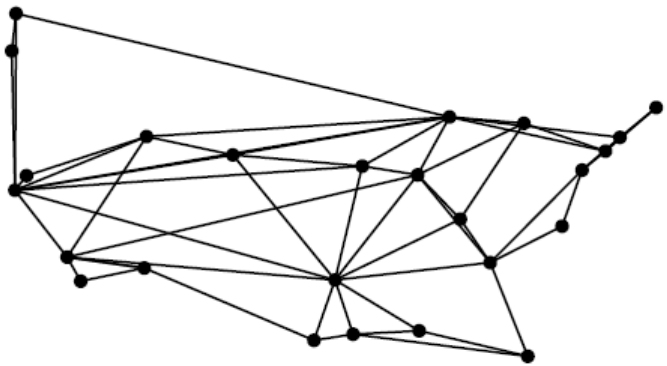
$$\le L^* + \left|\left\{a \in p : L_a^I > 0\right\}\right| \quad \text{(By Lemma)}$$

$$\le L^I + \sum_{a \in p} L_a^I$$

$$\le 2L^I$$

■

Experimental Results



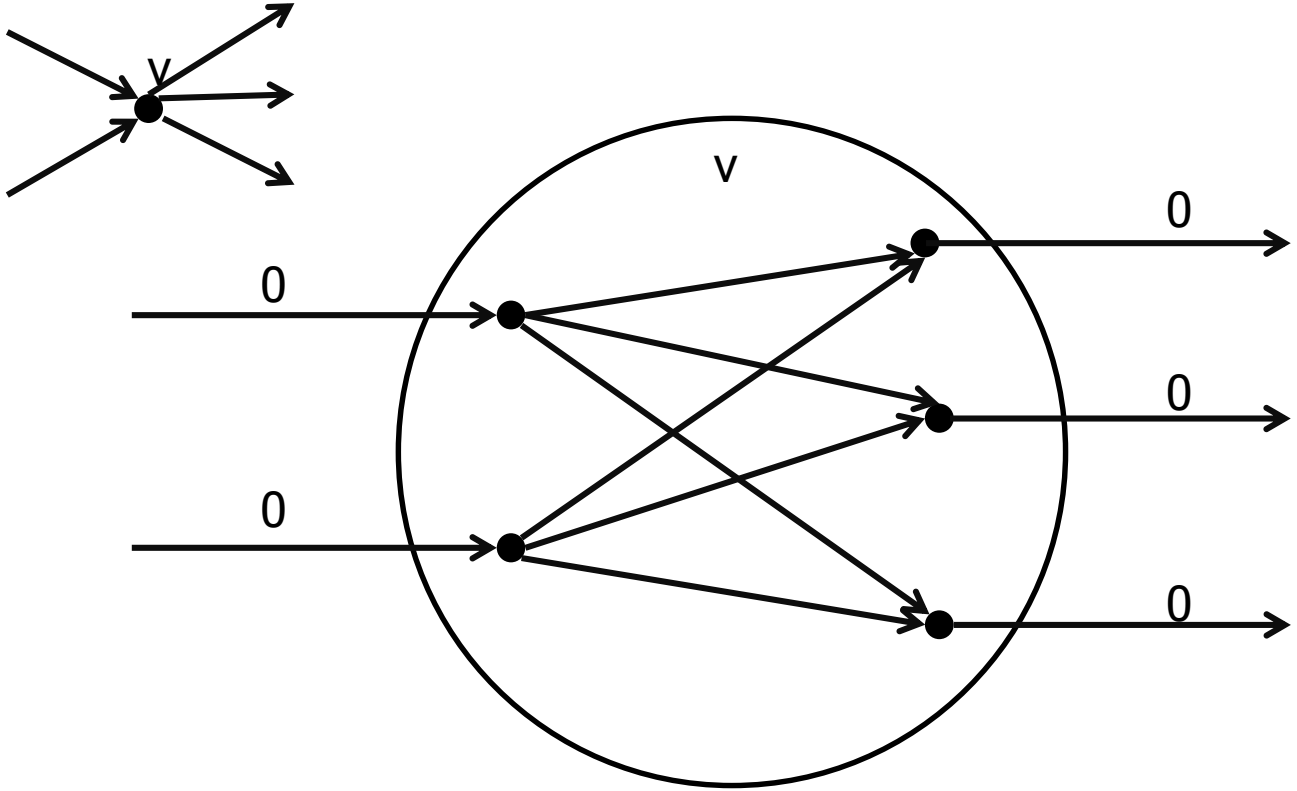
- AT&T MPLS backbone network (2008), 25 vertices, 224 arcs
- 600 shortest paths between all pairs of vertices
- Fixed-length encoding has maximal length 15 bits
- Proposed variable-length encoding has maximal length 10 bits
- More than 30% reduction in encoded path length

Gradient Descent Algorithm

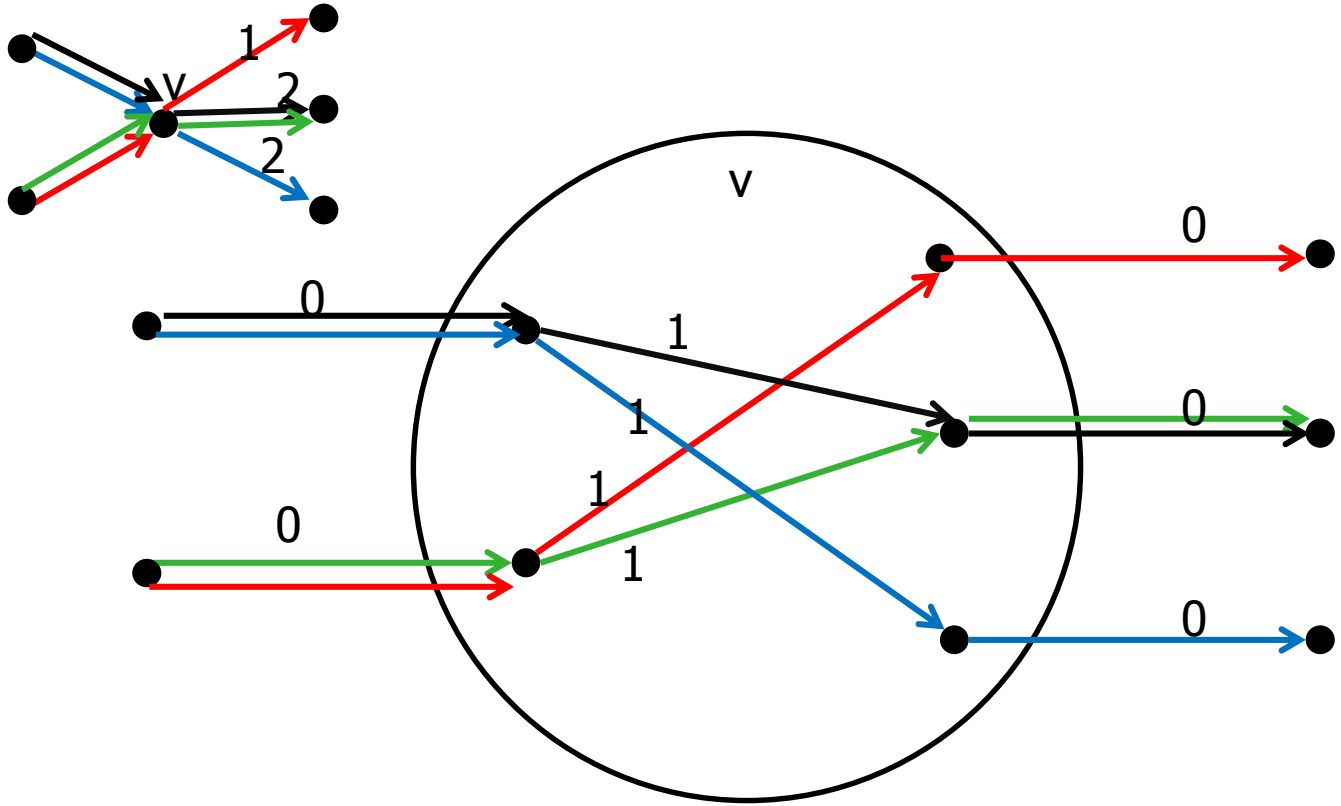
- An entropy form of the dual can be obtained (see paper)
- Dual used to derive "simple" projected gradient descent algorithm

Graph	Fixed Length	GDA	Nodes	Edges	Paths
AT&T MPLS backbone network	15 bits	10 bits	25	224	600
AS 4323 (Rocketfuel database)	25 bits	17 bits	51	284	2550

Note 1: Finer Grained Model



Note 1: Finer Grained Model



Conclusions

- Introduced and formalized the optimal path-encoding problem
- Showed that optimal path encoding is APX-hard
- Presented a 2-approximation algorithm
- Experiments show > 30% reduction in longest path encoding in real networks

Future Work

- Close gap between 2-approx and 8/7 lower bound on approx.
- What about multicast?

Thank you!