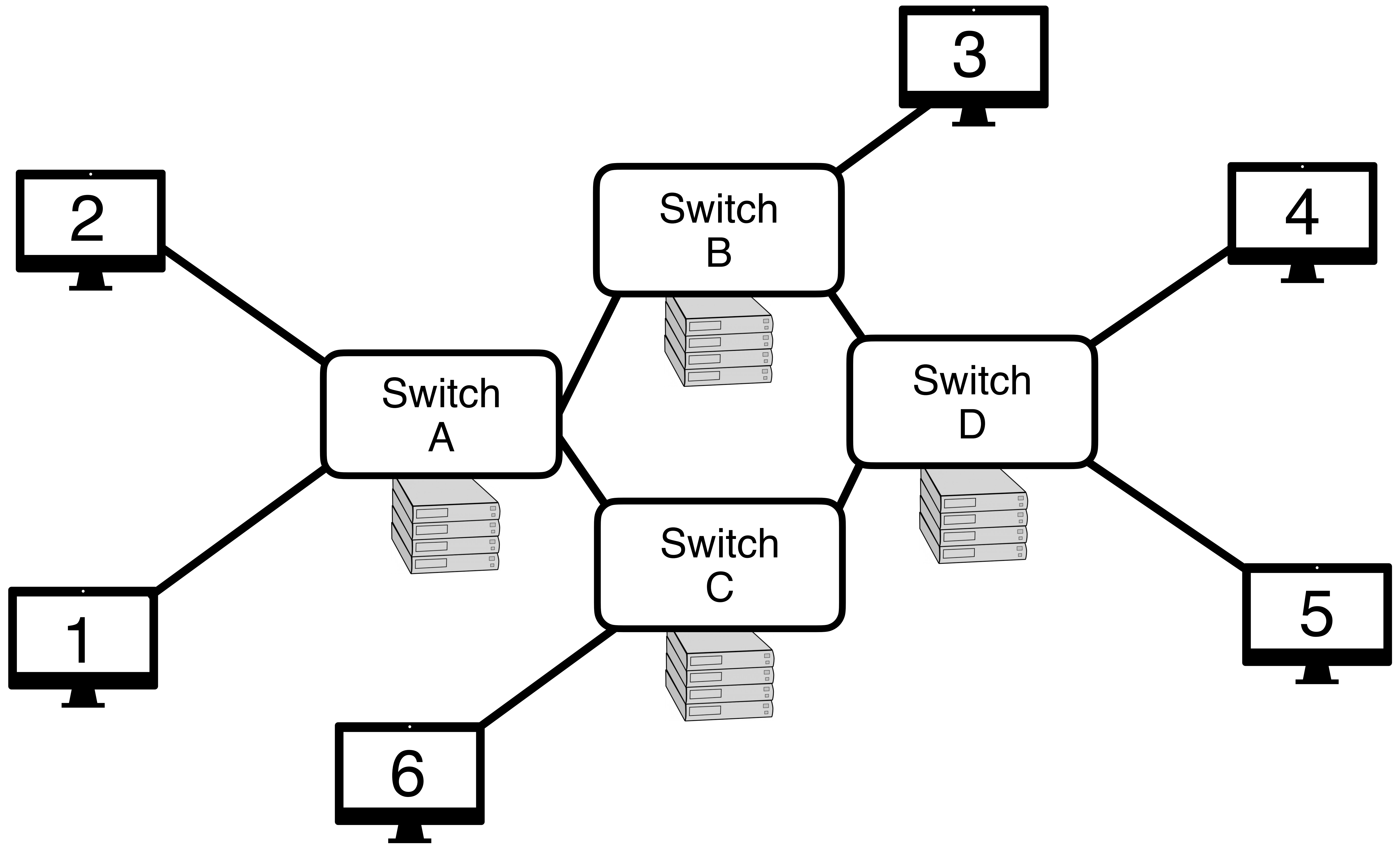
KATch: A Fast Symbolic Verifier for NetKAT

PLDI, June 2024

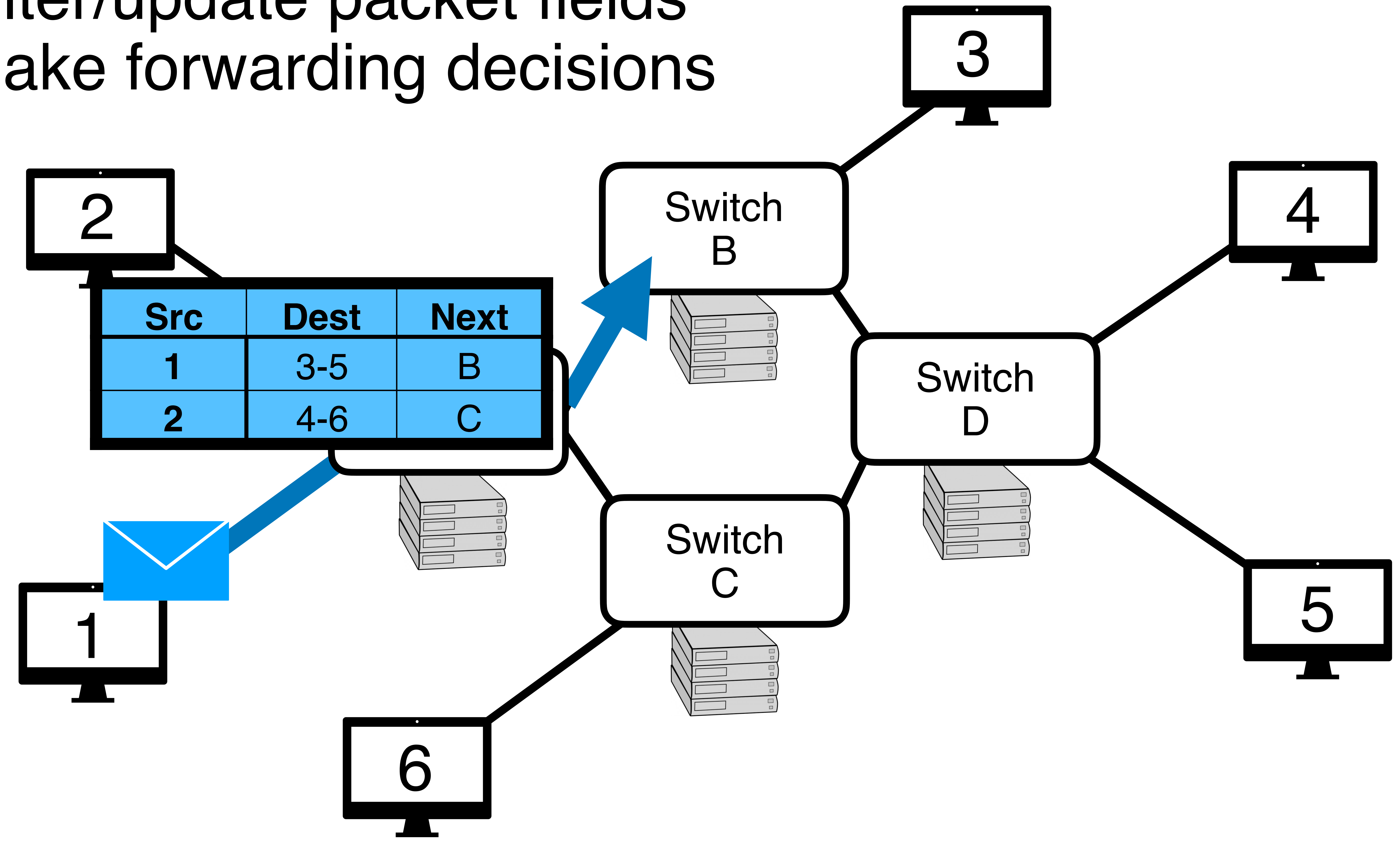
Mark Moeller
Jules Jacobs
Olivier Savary Belanger
David Darais
Cole Schlesinger
Steffen Smolka
Nate Foster
Alexandra Silva

Verification in Network Routing Policies



Verification in Network Routing Policies

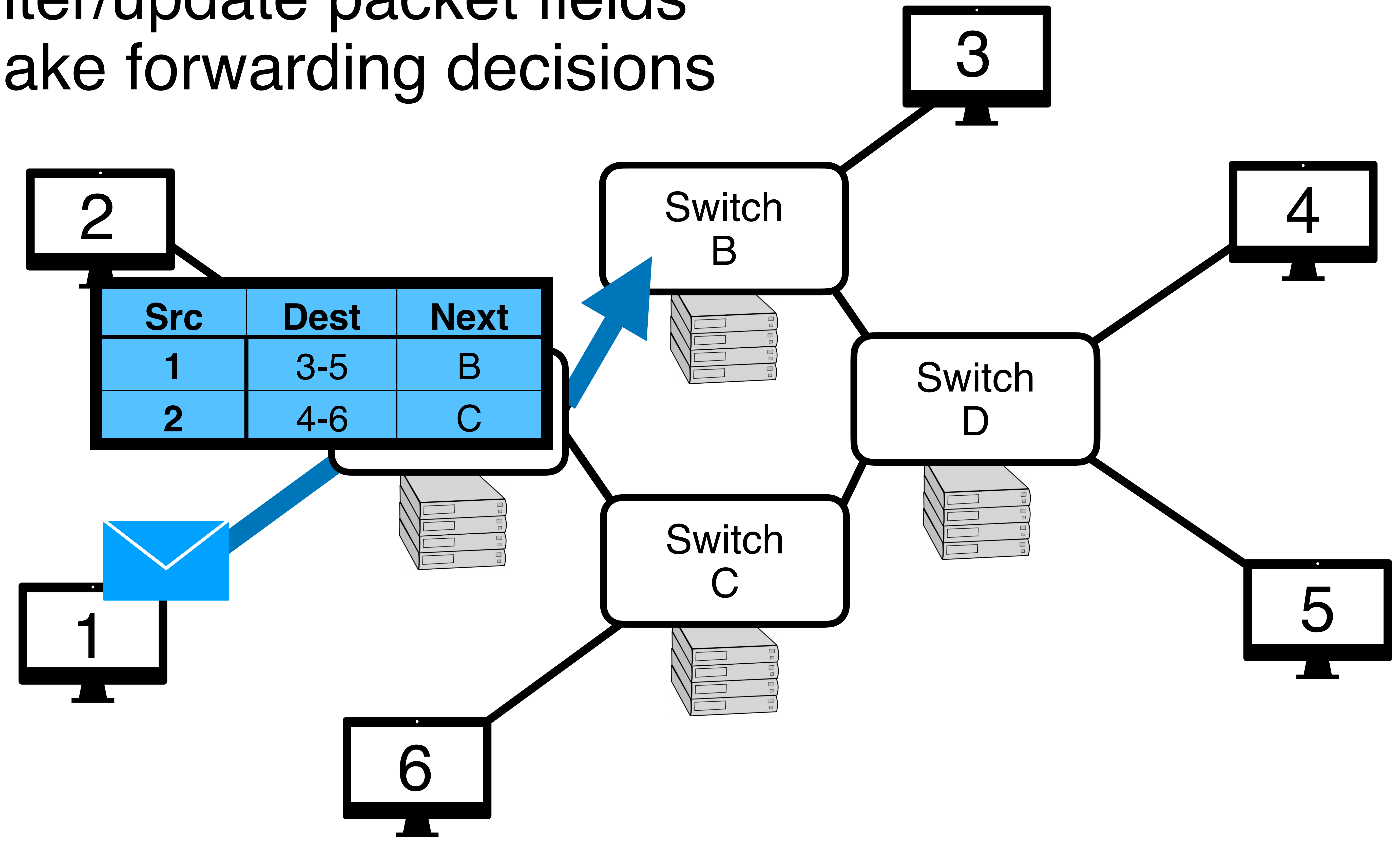
Forwarding policies:
1. Filter/update packet fields
2. Make forwarding decisions



Src	Dest	Next
1	3-5	B
2	4-6	C

Verification in Network Routing Policies

Forwarding policies:
1. Filter/update packet fields
2. Make forwarding decisions

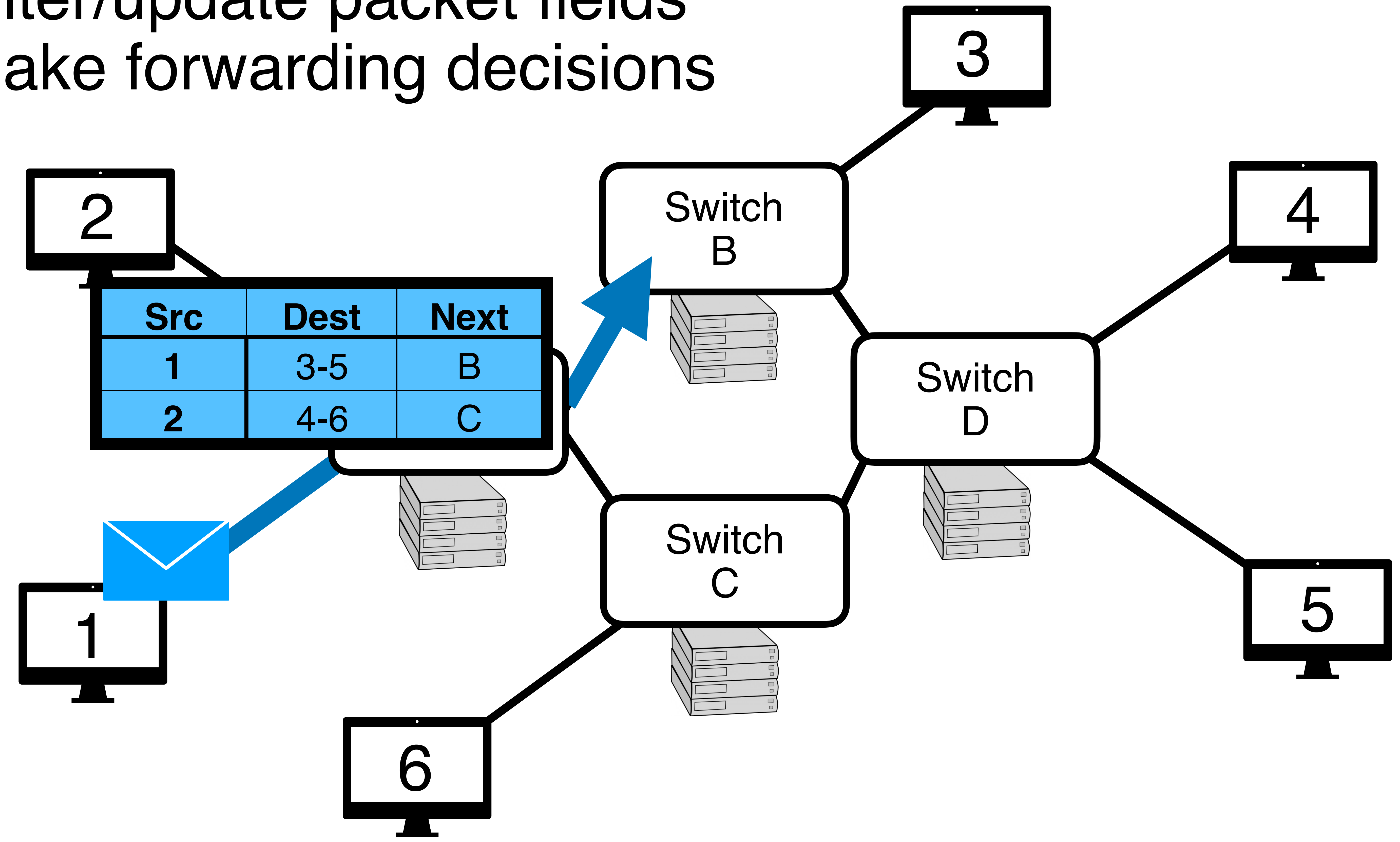


Src	Dest	Next
1	3-5	B
2	4-6	C

Verification questions:

Verification in Network Routing Policies

Forwarding policies:
1. Filter/update packet fields
2. Make forwarding decisions



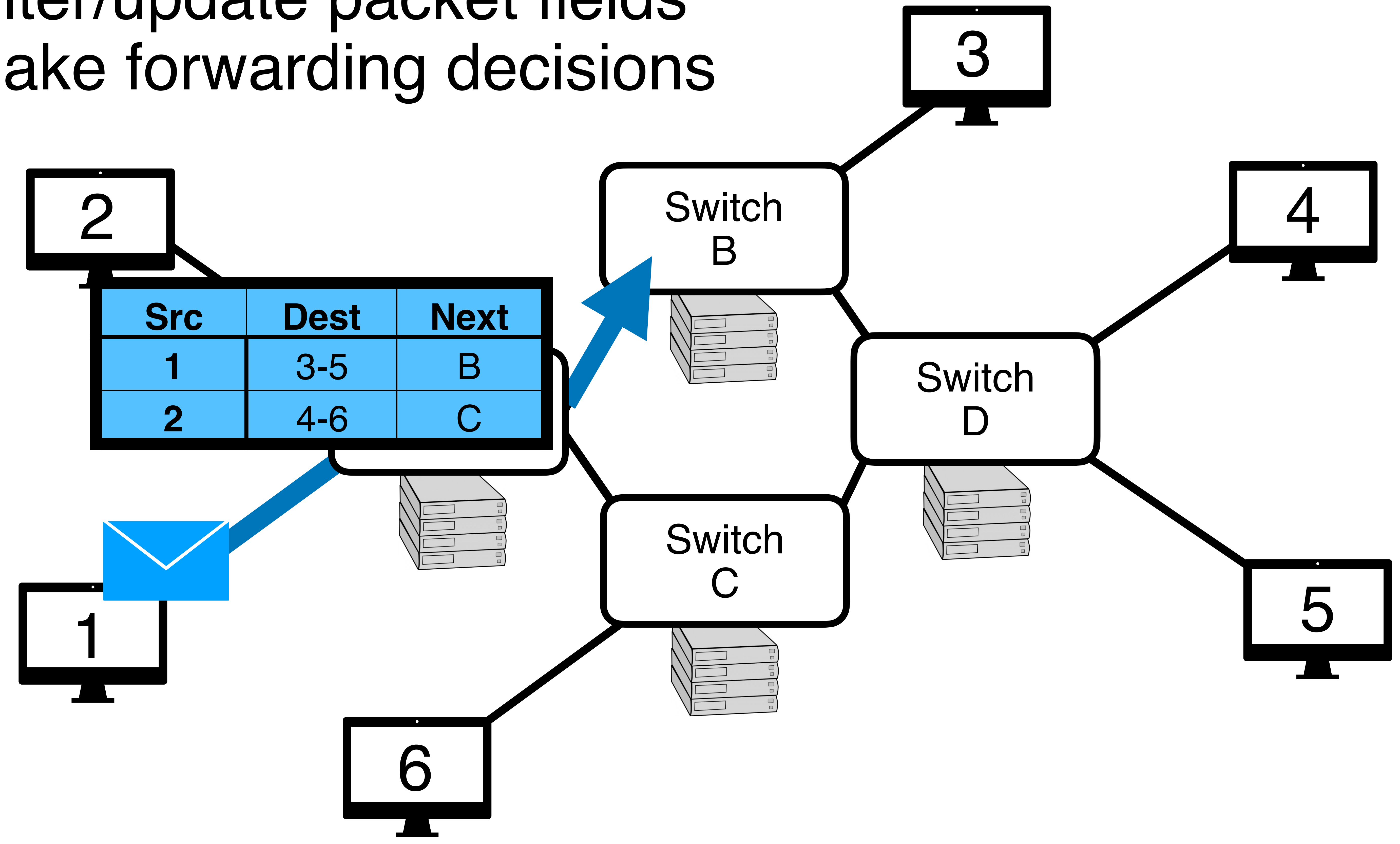
Src	Dest	Next
1	3-5	B
2	4-6	C

Verification questions:

Are all hosts reachable from every other host?

Verification in Network Routing Policies

Forwarding policies:
1. Filter/update packet fields
2. Make forwarding decisions



Src	Dest	Next
1	3-5	B
2	4-6	C

Verification questions:

Are all hosts reachable from every other host?

Are slices isolated as intended?

Network Verification with NetKAT (POPL 2014)

Network Verification with NetKAT (POPL 2014)

$$p, q ::= \bot \mid \top \mid f = v \mid f \neq v \mid f \leftarrow v \mid \mathrm{dup} \mid p + q \mid p \cdot q \mid p^\star$$

Network Verification with NetKAT (POPL 2014)

Filter packets

$$p, q ::= \bot \mid \top \mid \boxed{f = v \mid f \neq v} \mid f \leftarrow v \mid \mathrm{dup} \mid p + q \mid p \cdot q \mid p^\star$$

Network Verification with NetKAT (POPL 2014)

Filter packets

$$p, q ::= \bot \mid \top \mid f = v \mid f \neq v \mid f \leftarrow v \mid \mathrm{dup} \mid p + q \mid p \cdot q \mid p^{\star}$$

Modify packets

Network Verification with NetKAT (POPL 2014)

Filter packets

$$p, q ::= \bot \mid \top \mid f = v \mid f \neq v \mid f \leftarrow v \mid \mathrm{dup} \mid p + q \mid p \cdot q \mid p^{\star}$$

Modify packets

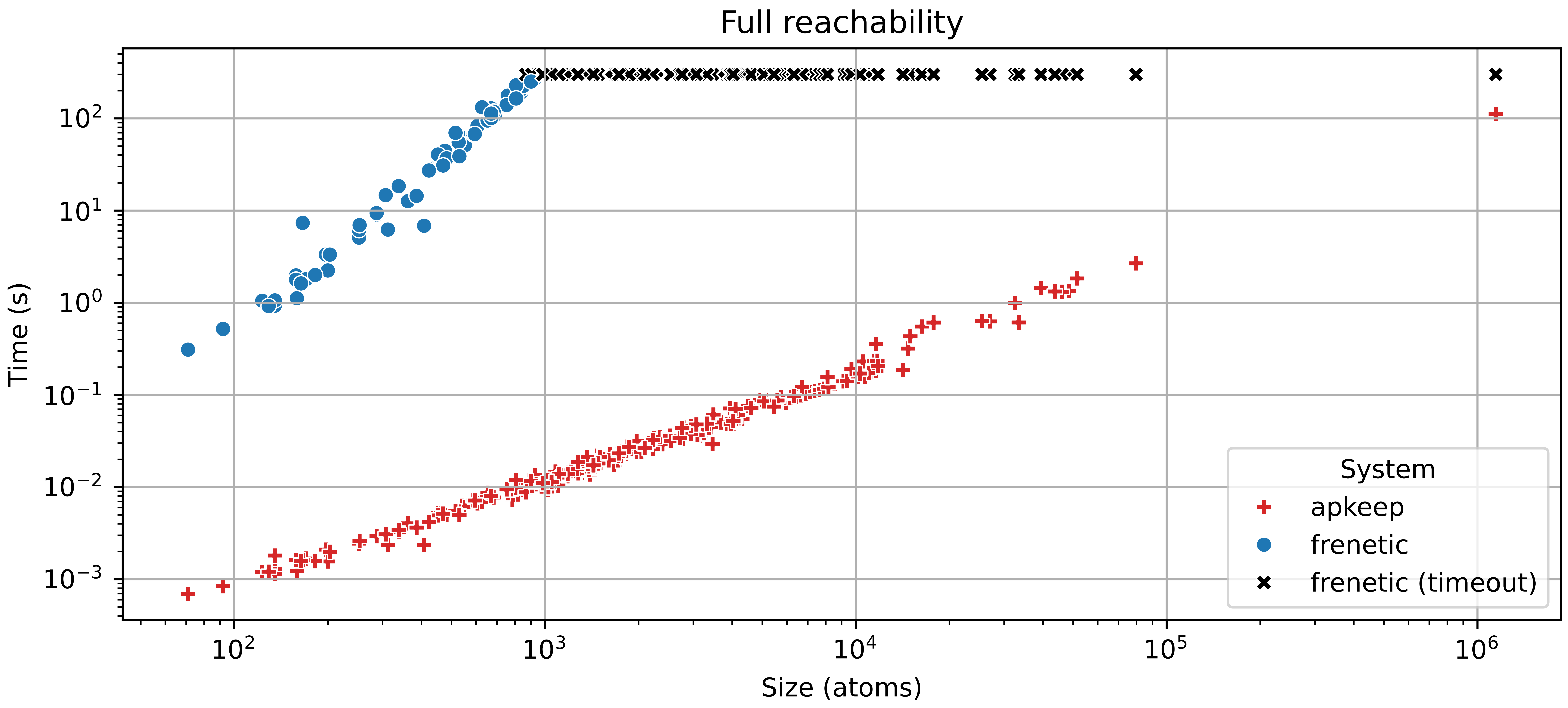
- NetKAT is sound, complete, and decidable

Network Verification with NetKAT (POPL 2014)

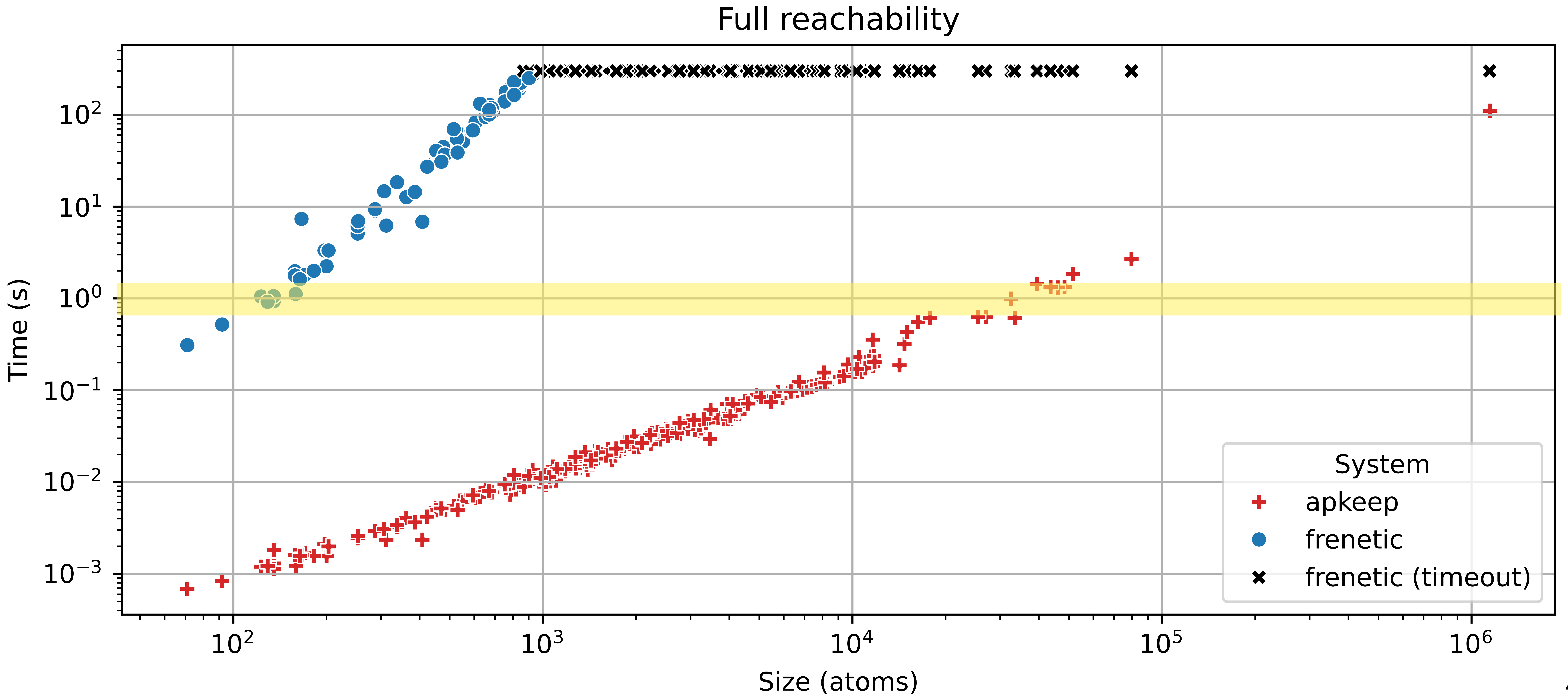
$$p, q ::= \bot \mid \top \mid \overbrace{f = v \mid f \neq v}^{\text{Filter packets}} \mid \underbrace{f \leftarrow v}_{\text{Modify packets}} \mid \mathrm{dup} \mid p + q \mid p \cdot q \mid p^\star$$

- NetKAT is sound, complete, and decidable
- Program equivalence is automata equivalenc

NetKAT and APKeep (NSDI 2020)



NetKAT and APKeep (NSDI 2020)



Problem:
NetKAT is limited in practice

Contributions (This work, PLDI 2024):

1. Symbolic packets and techniques

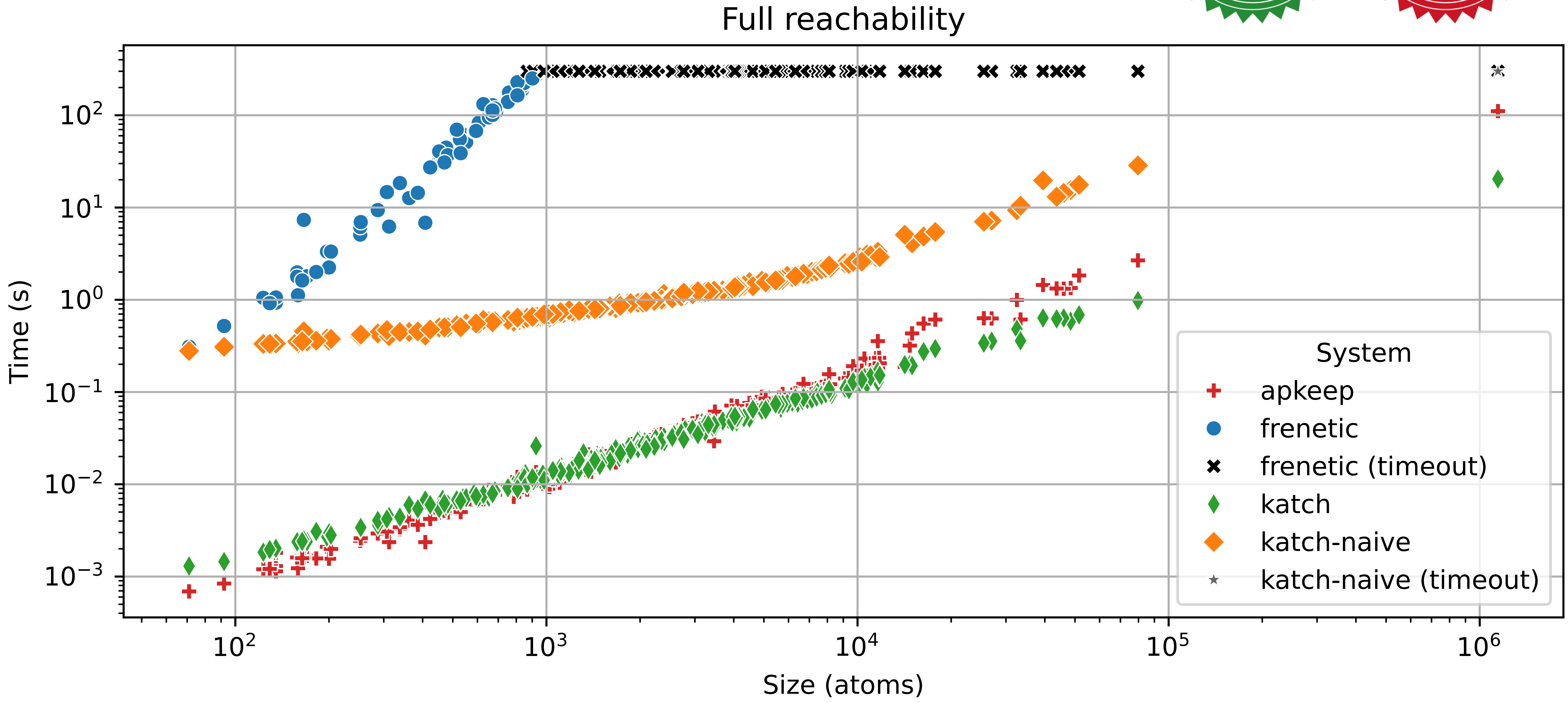
2. Extended NetKAT language

3. Symbolic counterexamples

And it is performant!

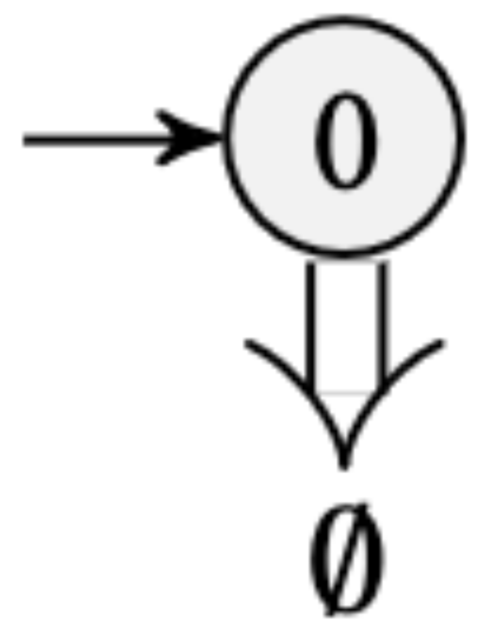
Artifacts Available acm V1.1

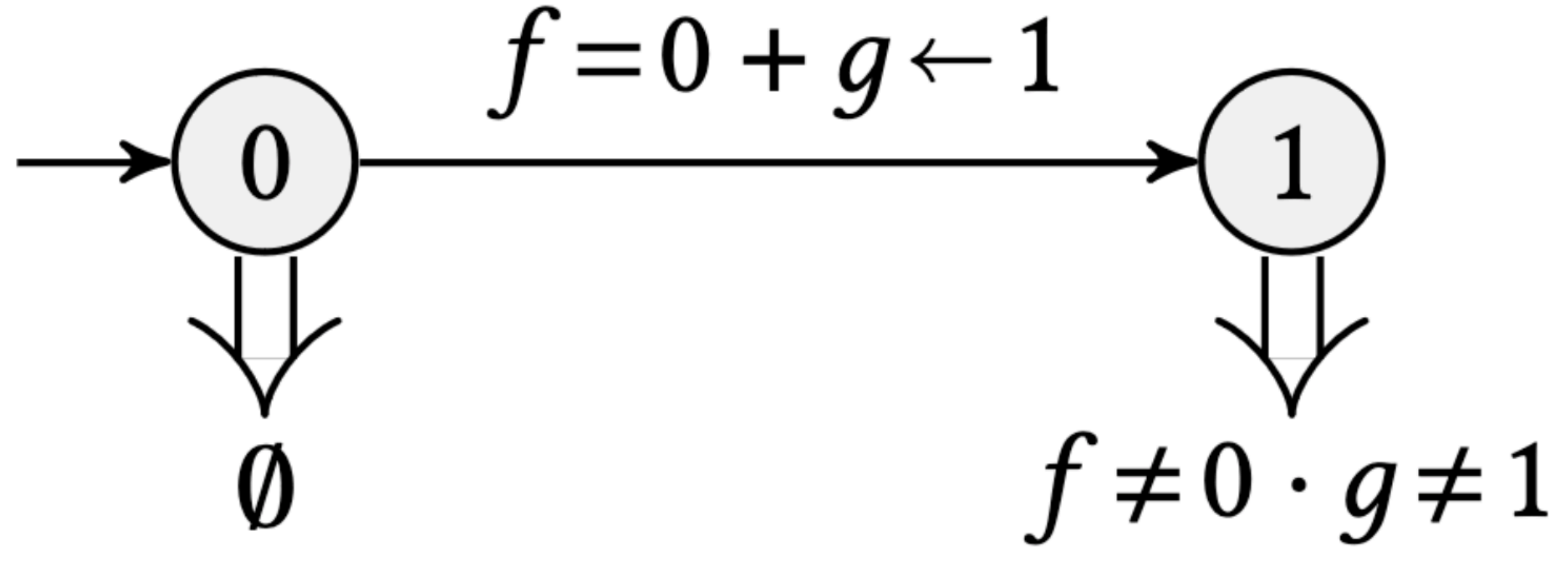
Artifacts Evaluated acm Reusable V1.1



What’s hard about NetKAT equivalence?

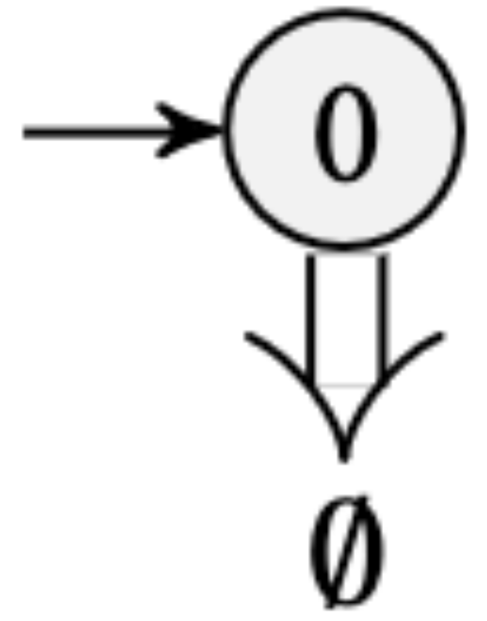
Are these two NetKAT automata equivalent?

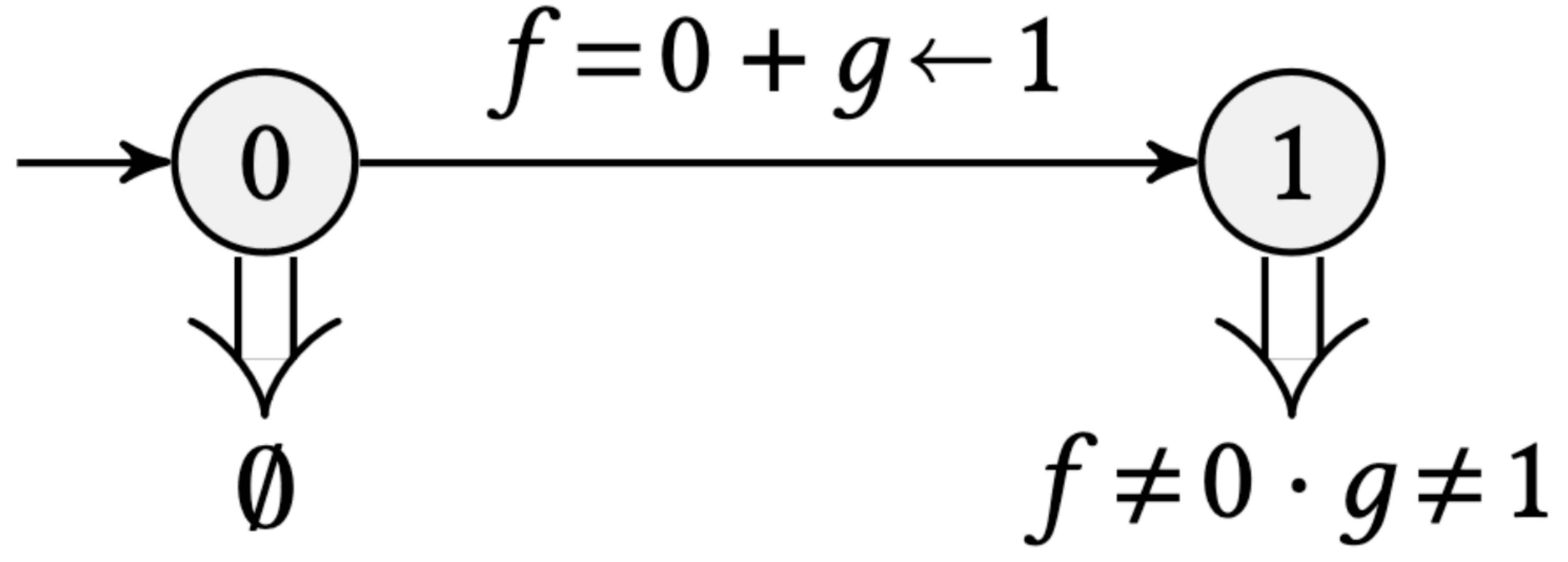




What's hard about NetKAT equivalence?

Are these two NetKAT automata equivalent?





S is the set of states,
Pk is the set of all packets $\epsilon : S \times \mathsf{Pk} \to 2^{\mathsf{Pk}}$

What’s hard about NetKAT equivalence?

Are these two NetKAT automata equivalent?

$$\delta : S \times \mathsf{Pk} \to S^{\mathsf{Pk}}$$

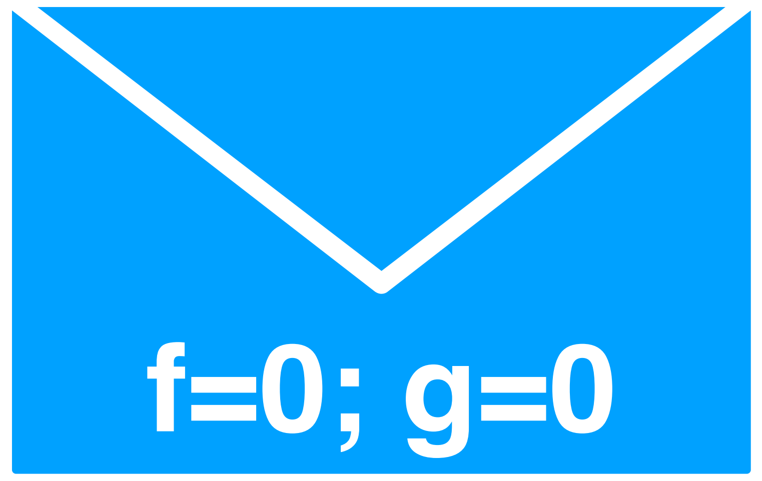
$$\begin{tikzcd}
{} \arrow[r] & 0 \arrow[d, Rightarrow] \\
 & \emptyset
\end{tikzcd}
\qquad\qquad
\begin{tikzcd}
{} \arrow[r] & 0 \arrow[r, "f=0+g\leftarrow 1"] \arrow[d, Rightarrow] & 1 \arrow[d, Rightarrow] \\
 & \emptyset & f\neq 0 \cdot g\neq 1
\end{tikzcd}$$

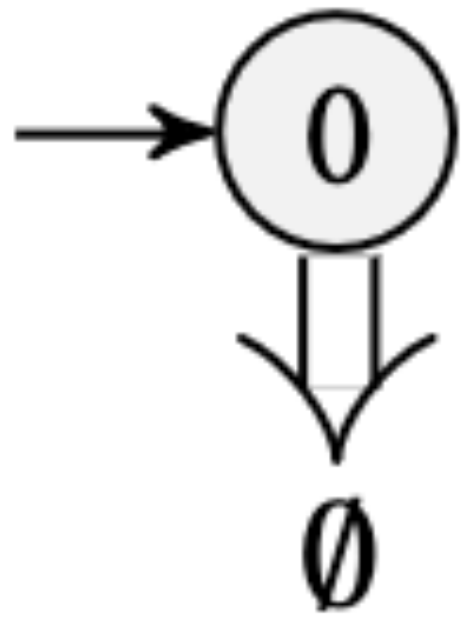
S is the set of states,
Pk is the set of all packets $\epsilon : S \times \mathsf{Pk} \to 2^{\mathsf{Pk}}$

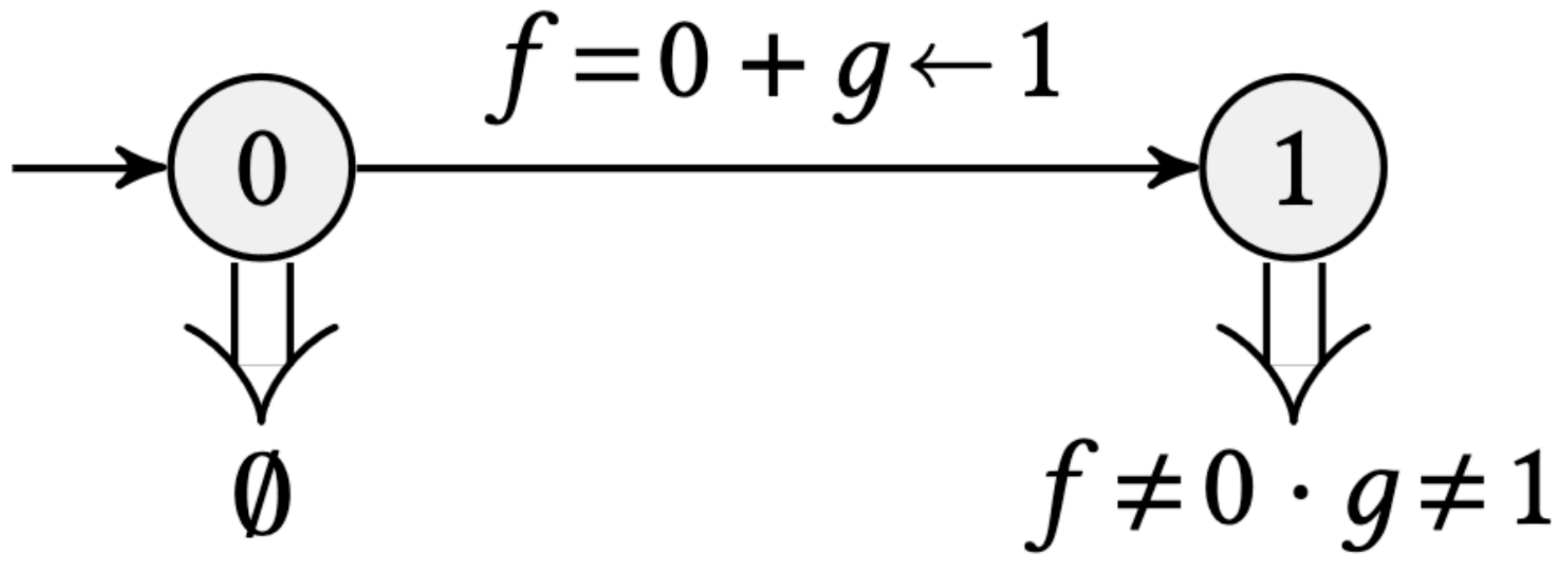
Wait! It’s not so simple…

Are these two NetKAT automata equivalent?

$$\delta : S \times \mathsf{Pk} \to S^{\mathsf{Pk}}$$





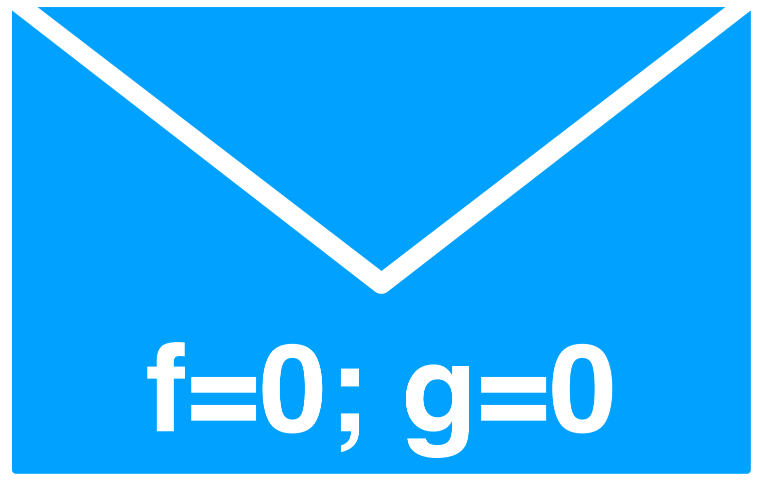


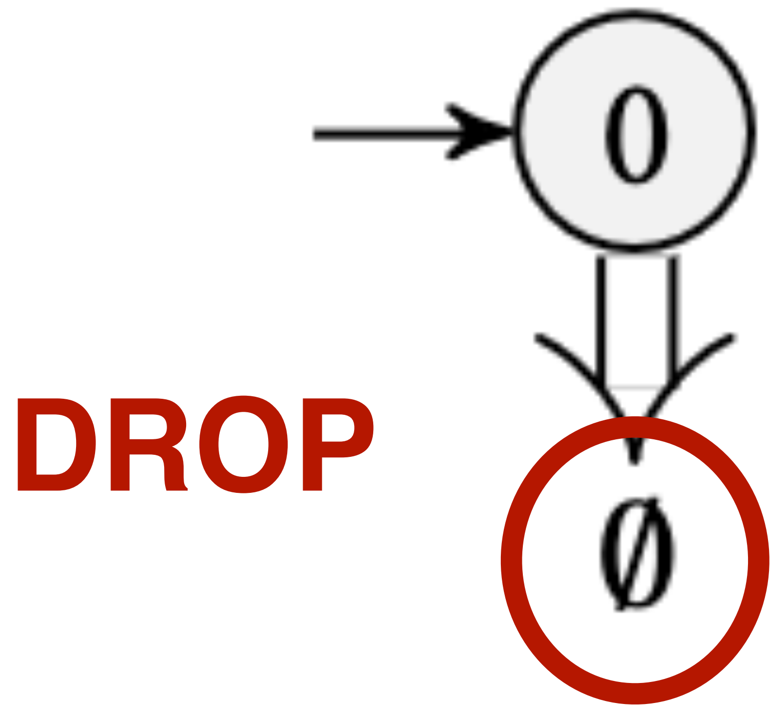
S is the set of states,
Pk is the set of all packets $\epsilon : S \times \mathsf{Pk} \to 2^{\mathsf{Pk}}$

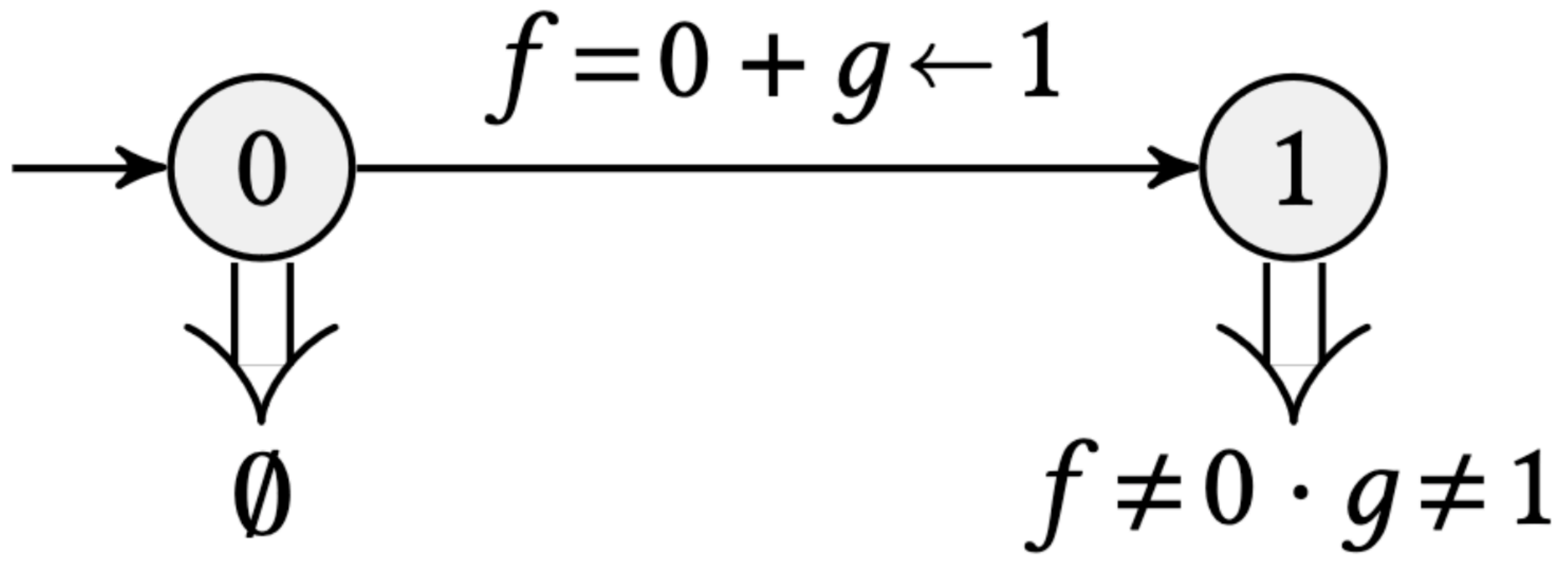
Wait! It's not so simple…

Are these two NetKAT automata equivalent?

$$\delta : S \times \mathrm{Pk} \to S^{\mathrm{Pk}}$$





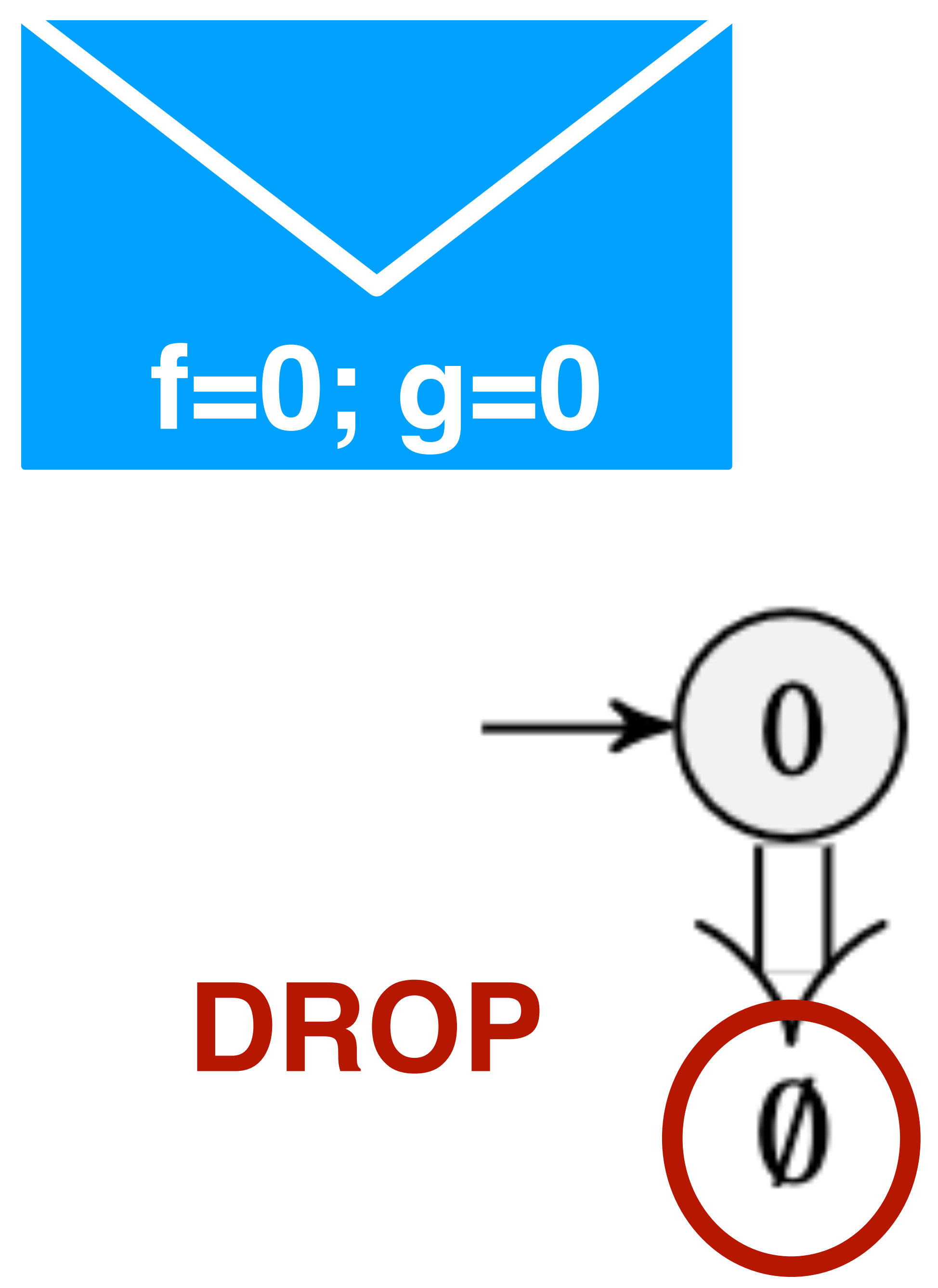


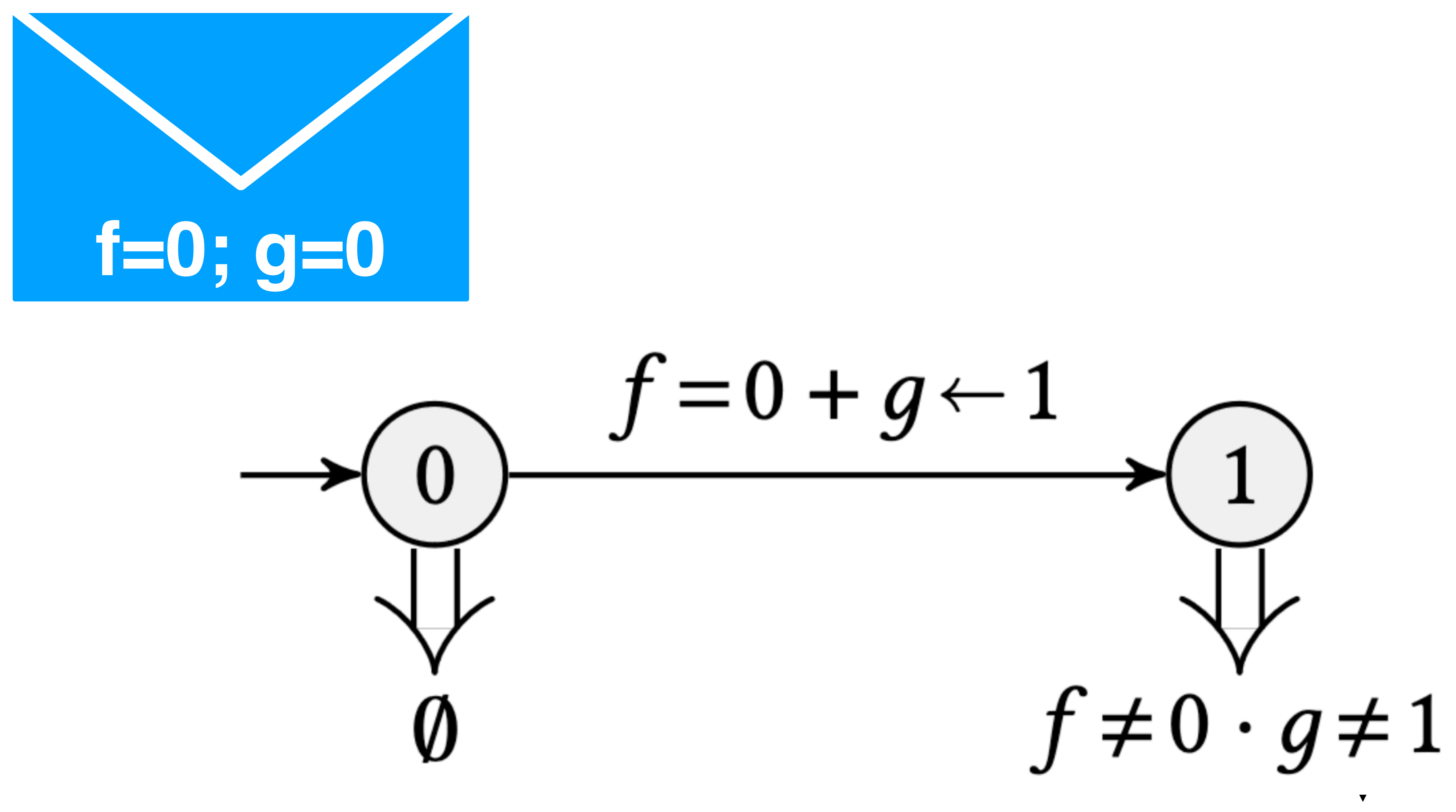
S is the set of states,
Pk is the set of all packets $\epsilon : S \times \mathrm{Pk} \to 2^{\mathrm{Pk}}$

Wait! It's not so simple…

Are these two NetKAT automata equivalent?

$\delta : S \times \mathsf{Pk} \to S^{\mathsf{Pk}}$



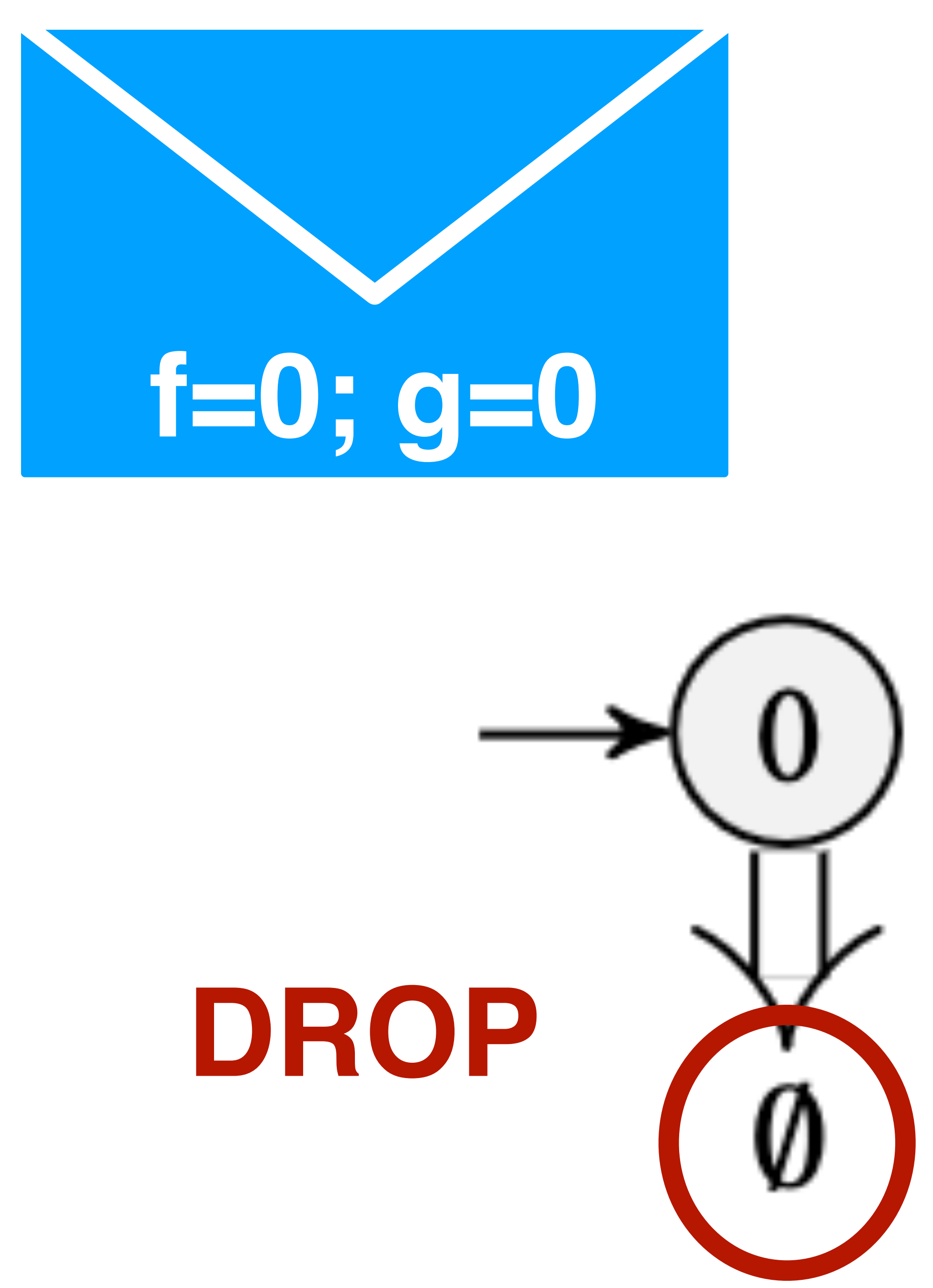


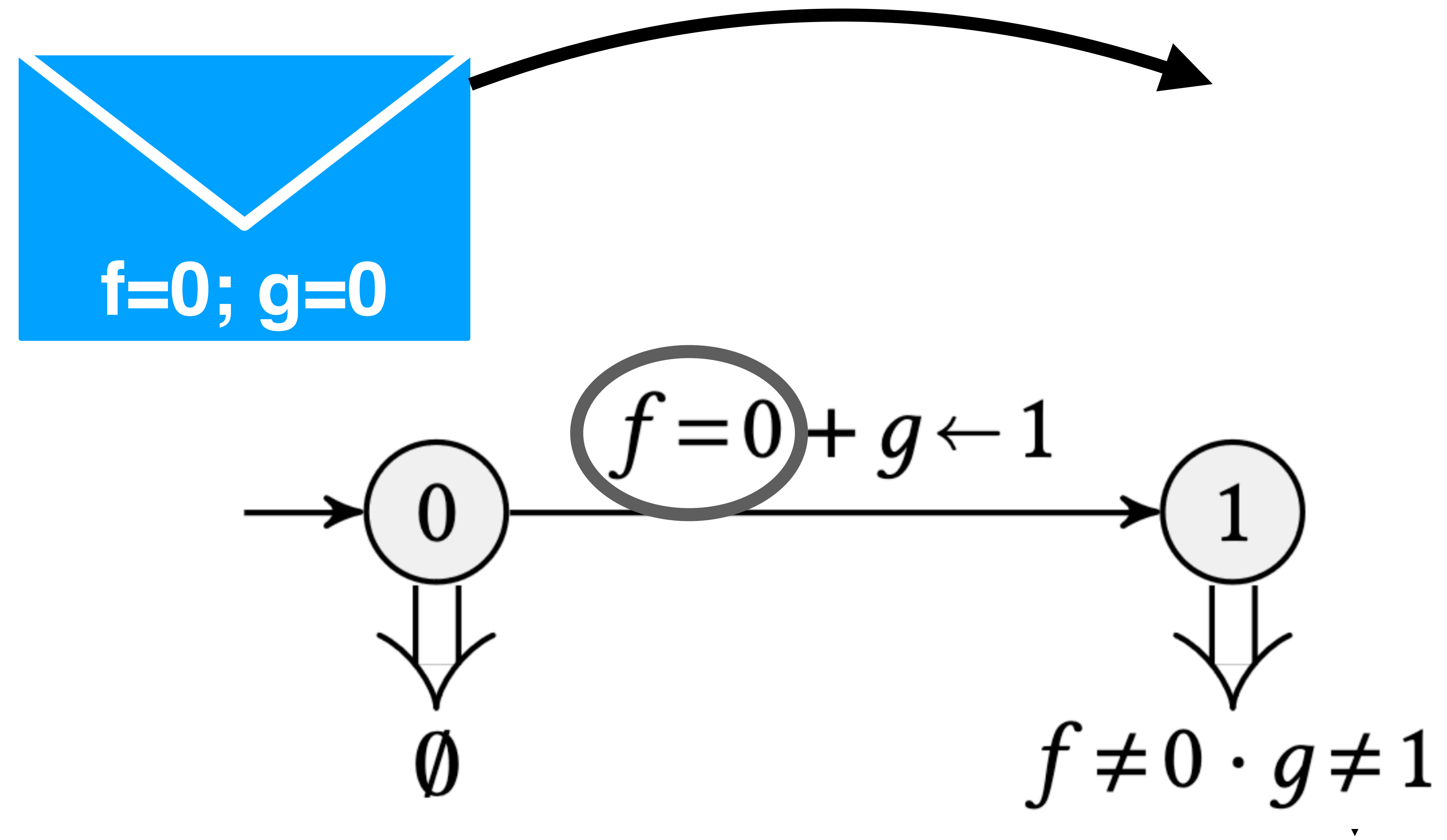
S is the set of states,
Pk is the set of all packets $\epsilon : S \times \mathsf{Pk} \to 2^{\mathsf{Pk}}$

Wait! It's not so simple…

Are these two NetKAT automata equivalent?

$\delta : S \times \mathsf{Pk} \to S^{\mathsf{Pk}}$



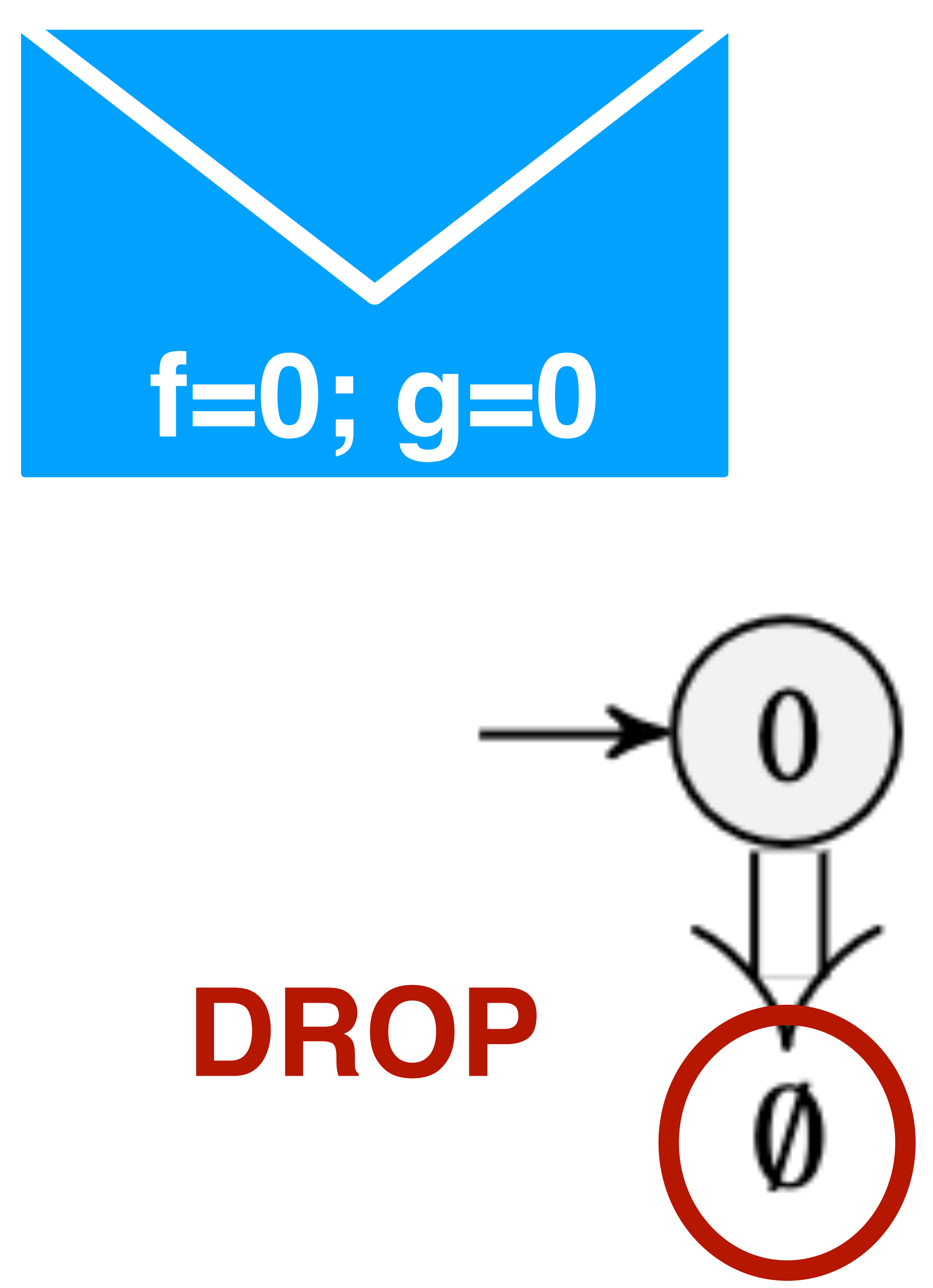


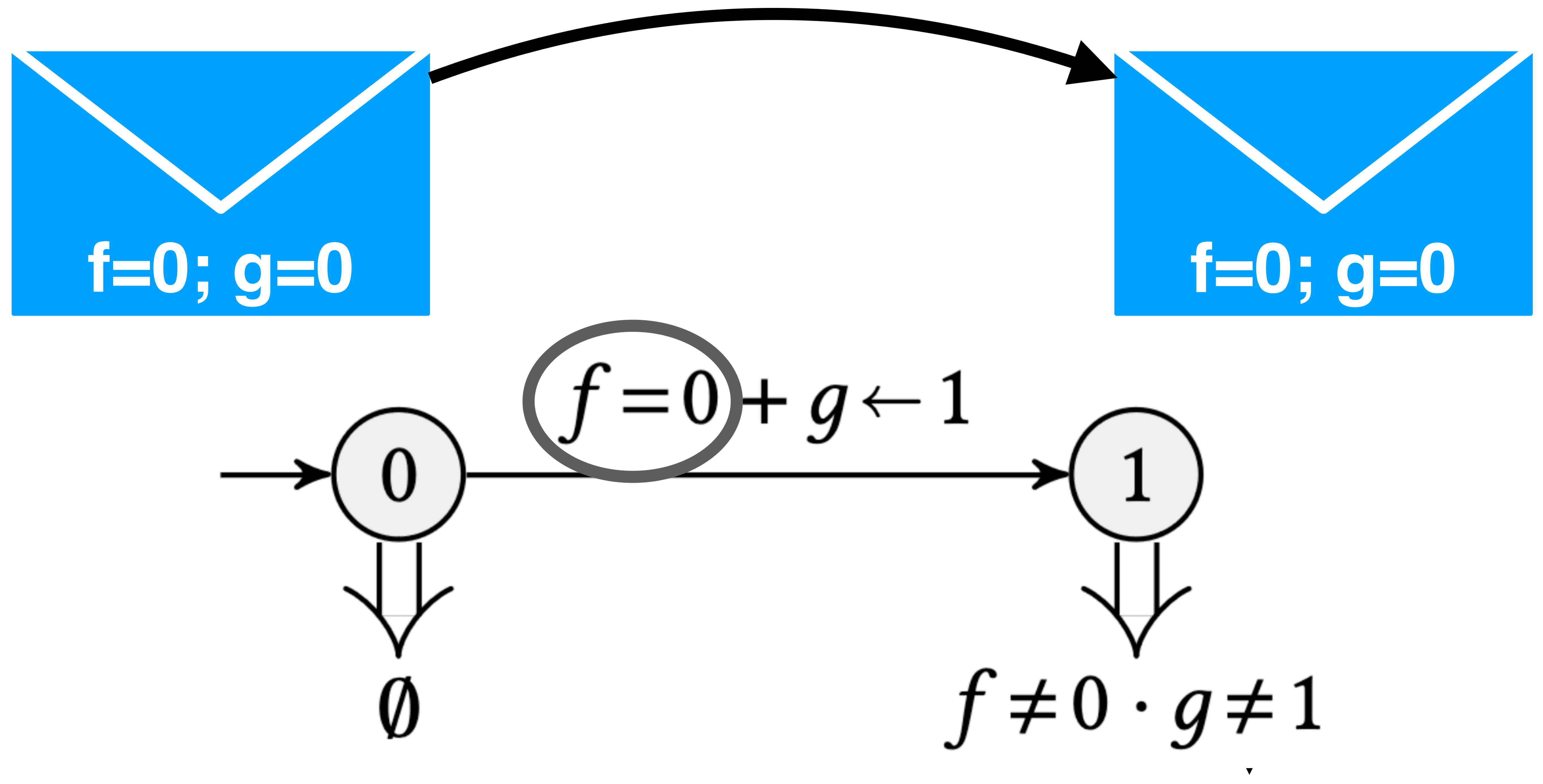
S is the set of states,
Pk is the set of all packets $\epsilon : S \times \mathsf{Pk} \to 2^{\mathsf{Pk}}$

Wait! It's not so simple…

Are these two NetKAT automata equivalent?

$\delta : S \times \mathsf{Pk} \to S^{\mathsf{Pk}}$



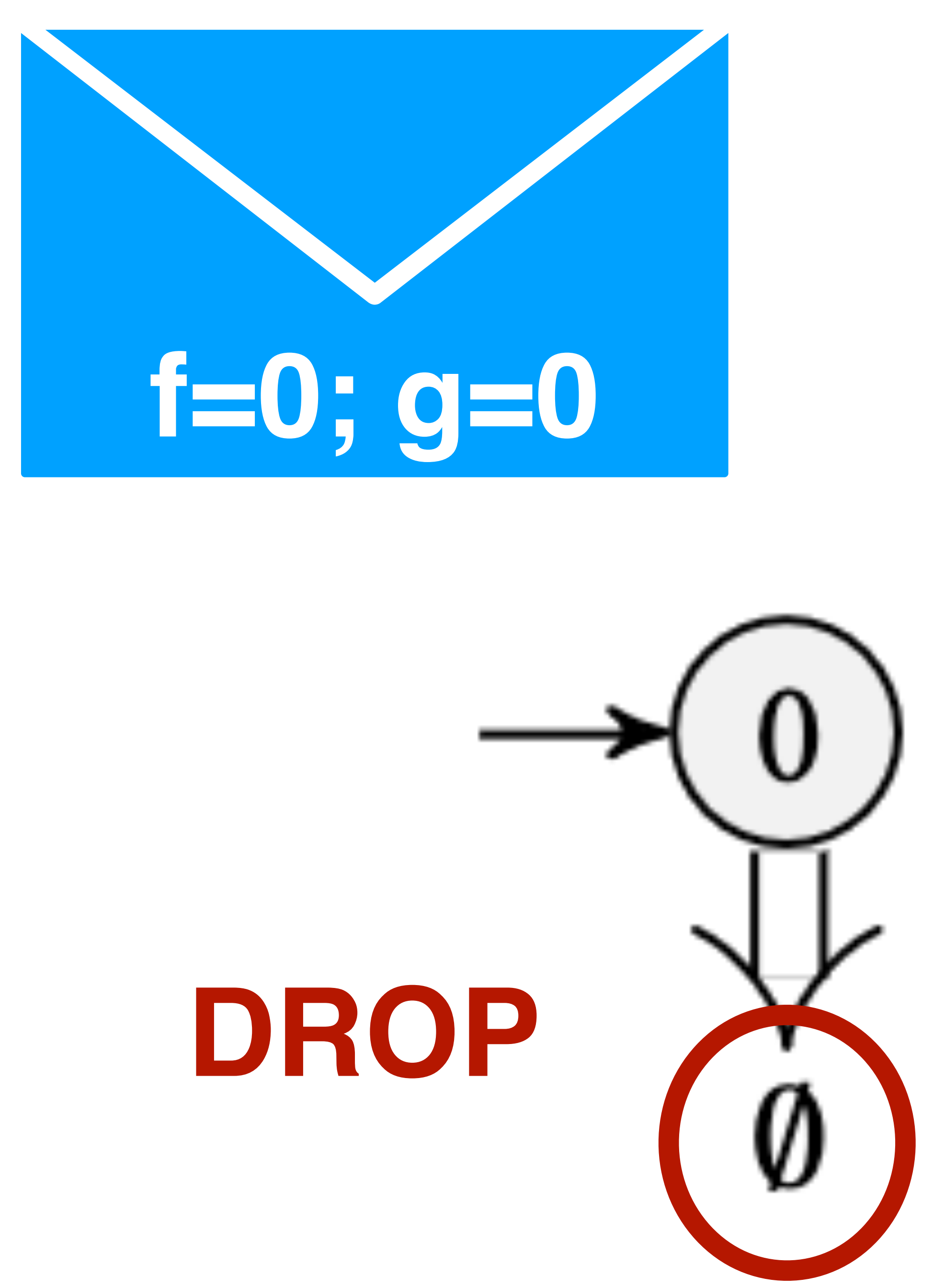


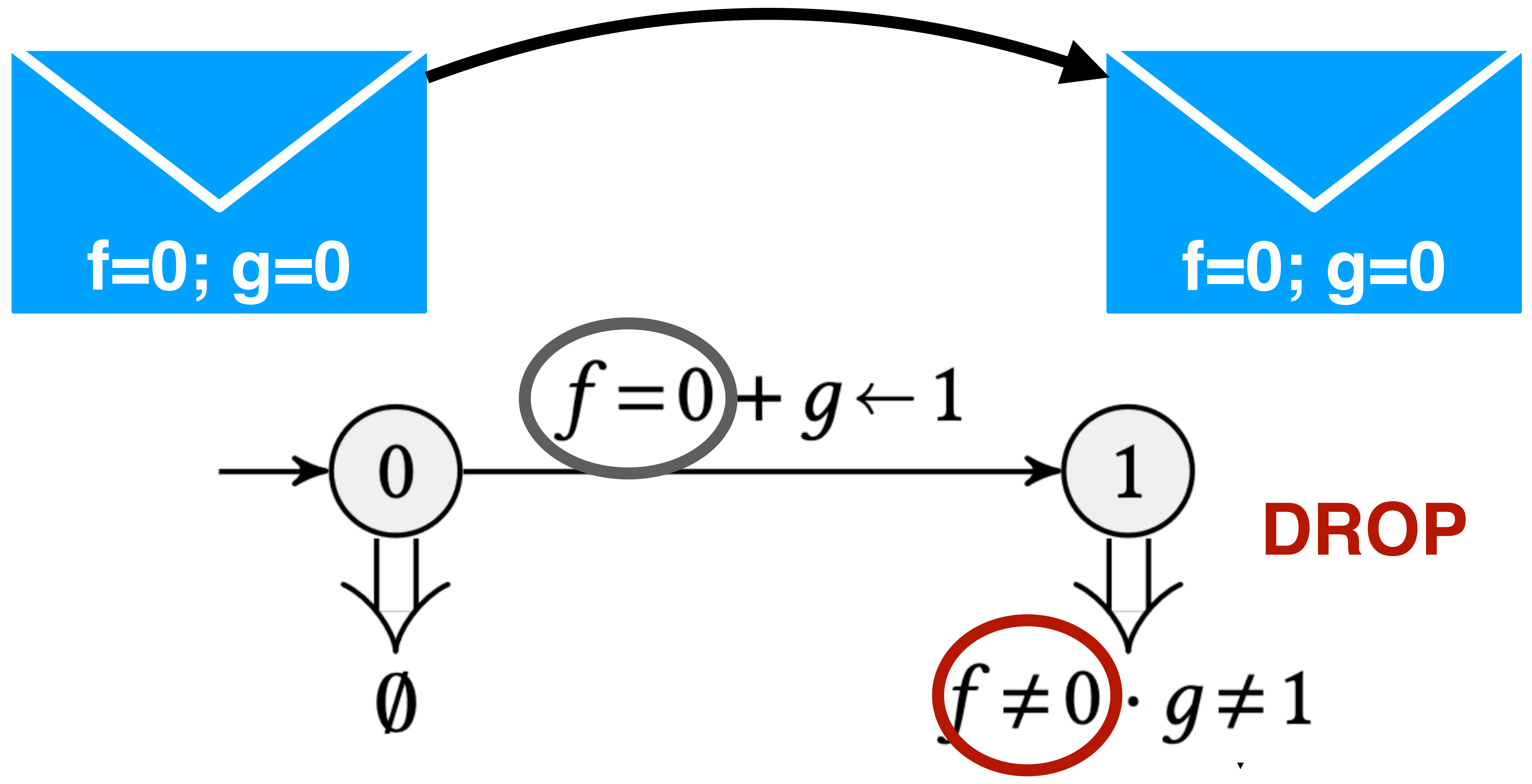
S is the set of states,
Pk is the set of all packets $\epsilon : S \times \mathsf{Pk} \to 2^{\mathsf{Pk}}$

Wait! It’s not so simple…

Are these two NetKAT automata equivalent?

$\delta : S \times \mathsf{Pk} \to S^{\mathsf{Pk}}$



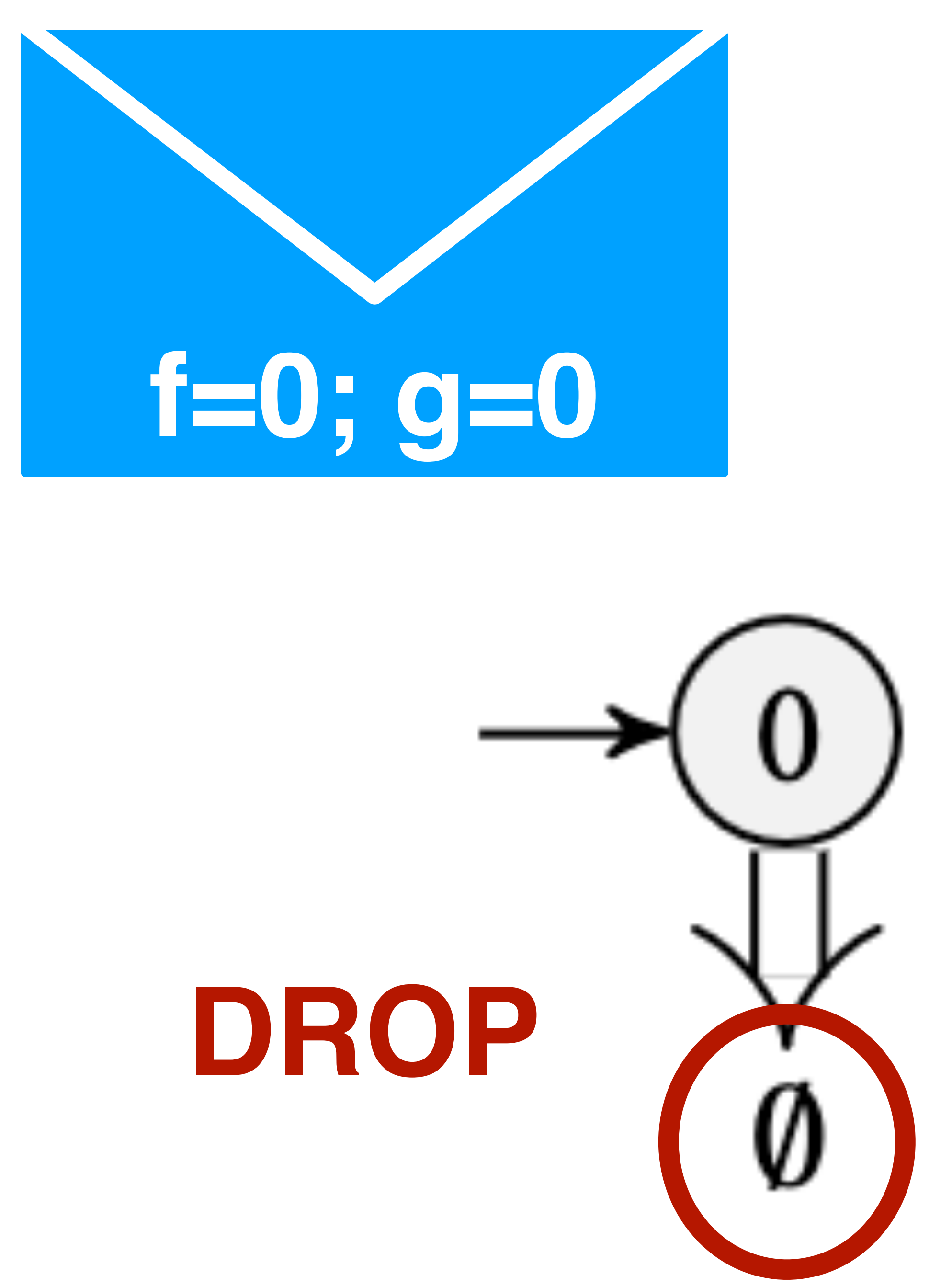


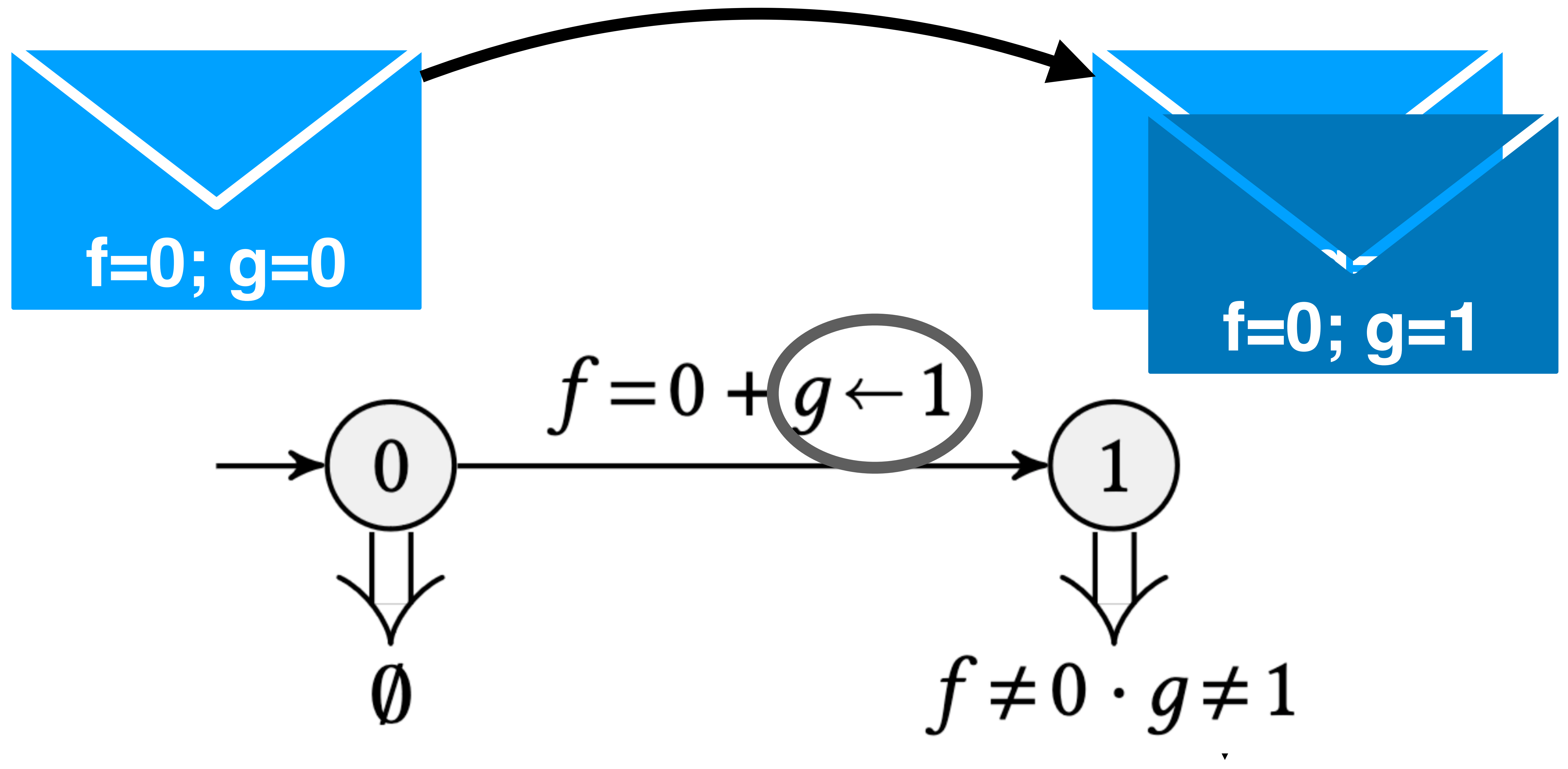
S is the set of states,
Pk is the set of all packets $\epsilon : S \times \mathsf{Pk} \to 2^{\mathsf{Pk}}$

Wait! It's not so simple…

Are these two NetKAT automata equivalent?

$\delta : S \times \mathsf{Pk} \to S^{\mathsf{Pk}}$





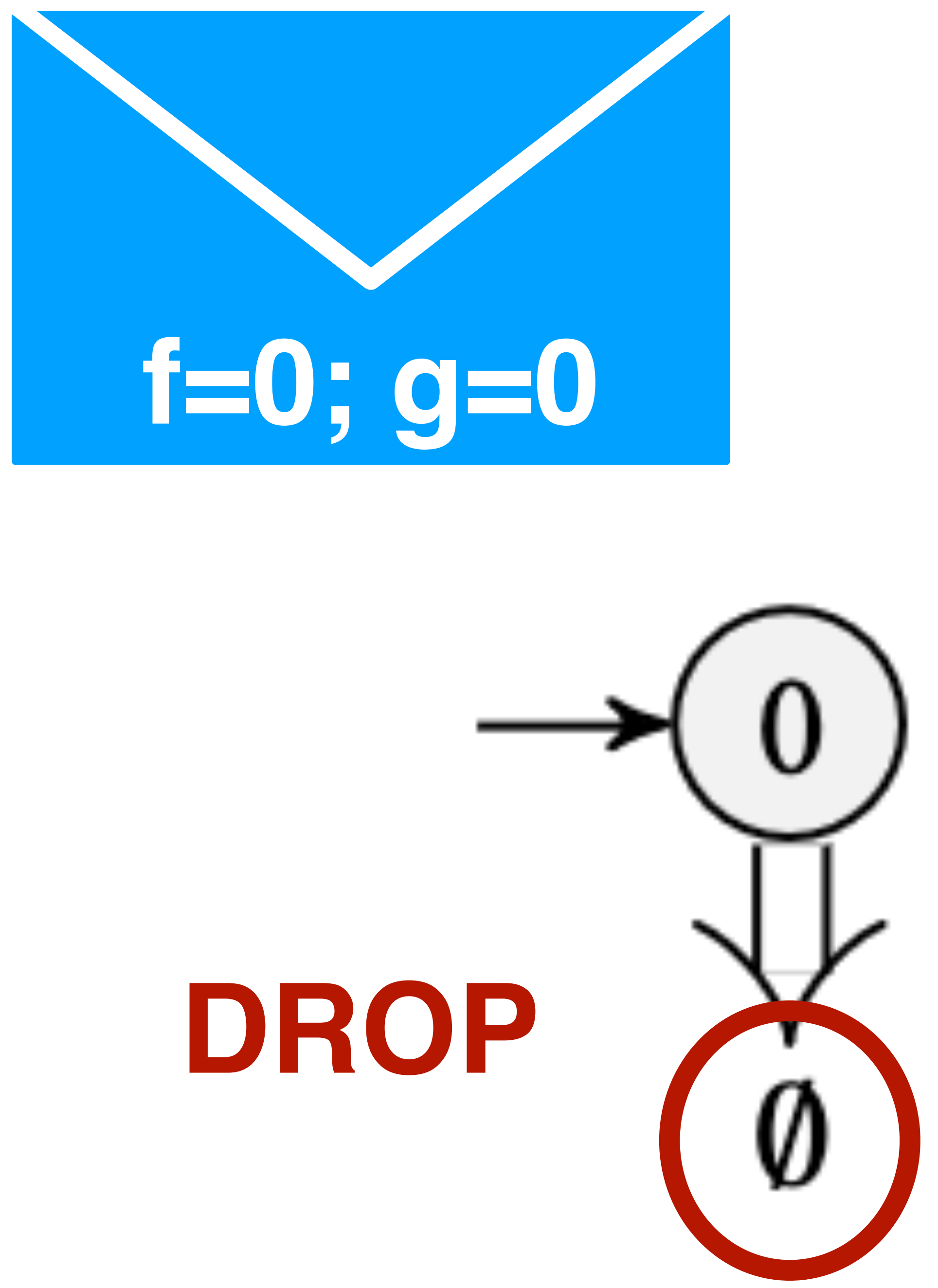
S is the set of states,
Pk is the set of all packets

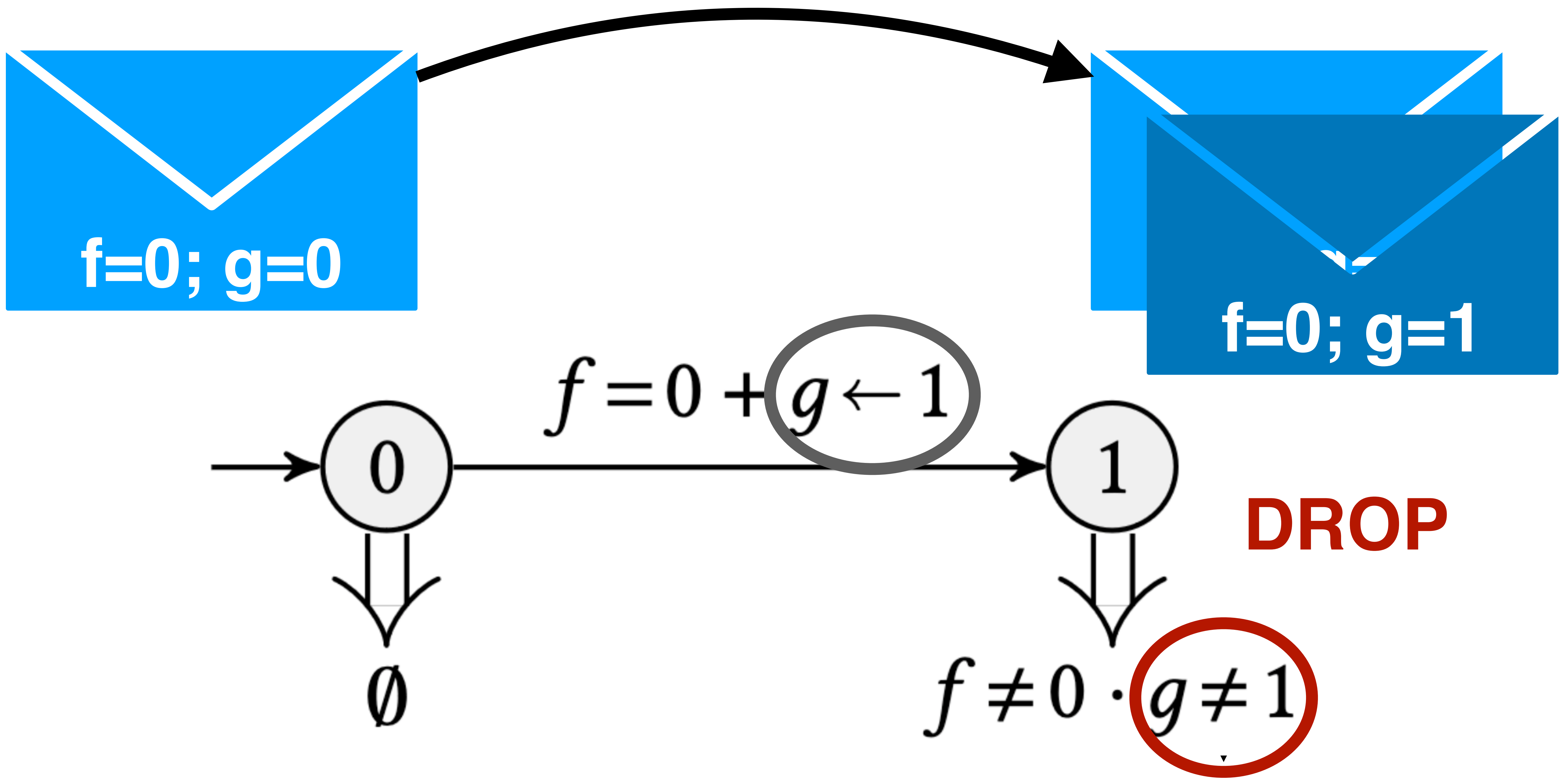
$\epsilon : S \times \mathsf{Pk} \to 2^{\mathsf{Pk}}$

Wait! It’s not so simple…

Are these two NetKAT automata equivalent?

$\delta : S \times \mathsf{Pk} \rightarrow S^{\mathsf{Pk}}$





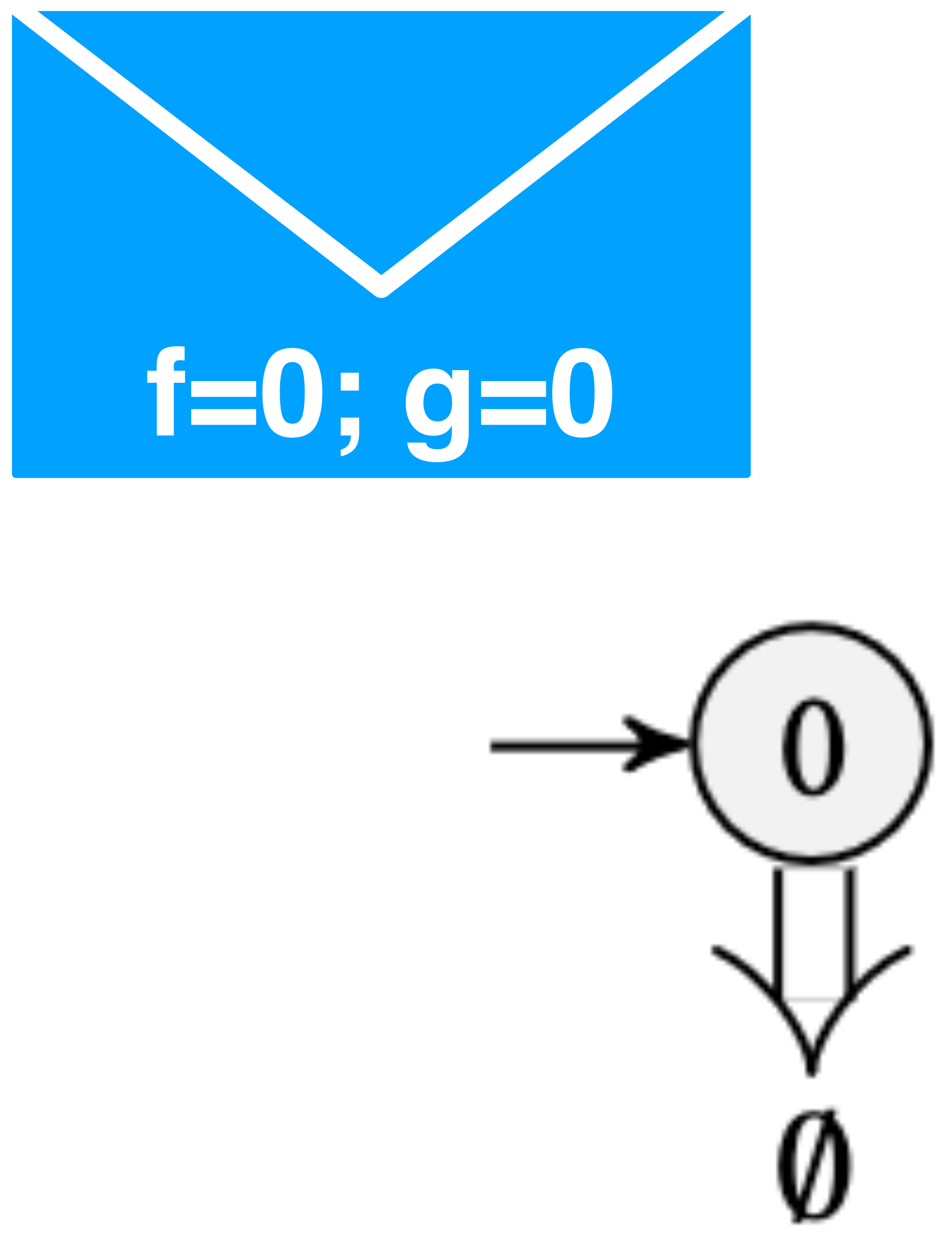
S is the set of states,
Pk is the set of all packets

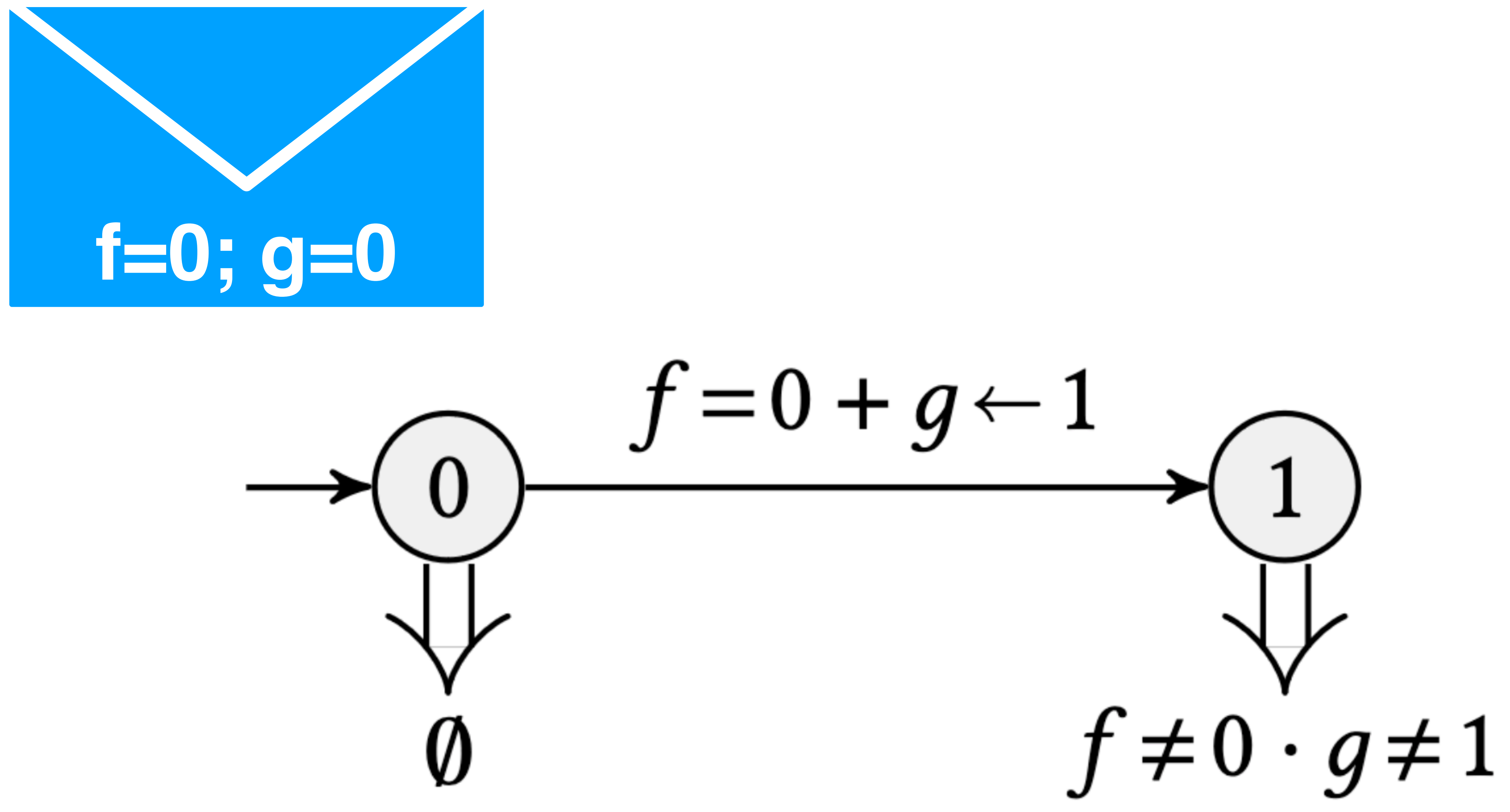
$\epsilon : S \times \mathsf{Pk} \rightarrow 2^{\mathsf{Pk}}$

Checking Equivalence in NetKAT

Are these two NetKAT automata equivalent?

$\delta : S \times \mathsf{Pk} \to S^{\mathsf{Pk}}$



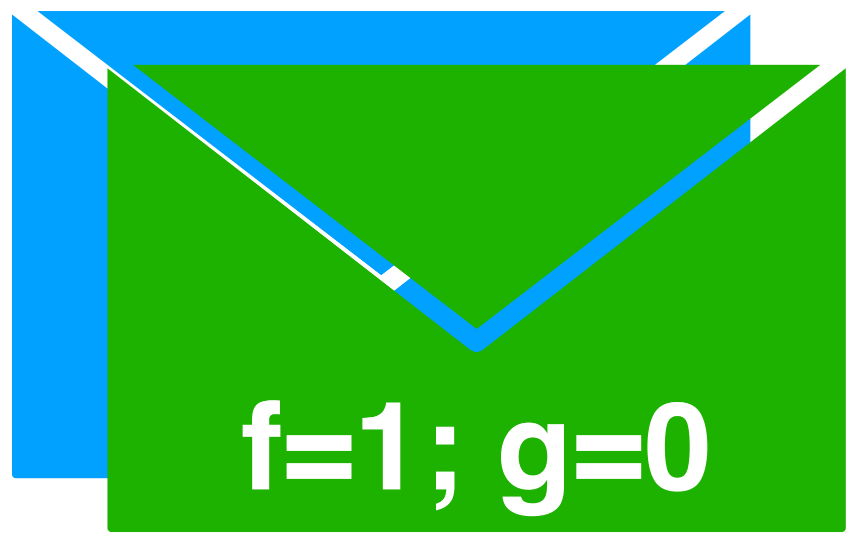


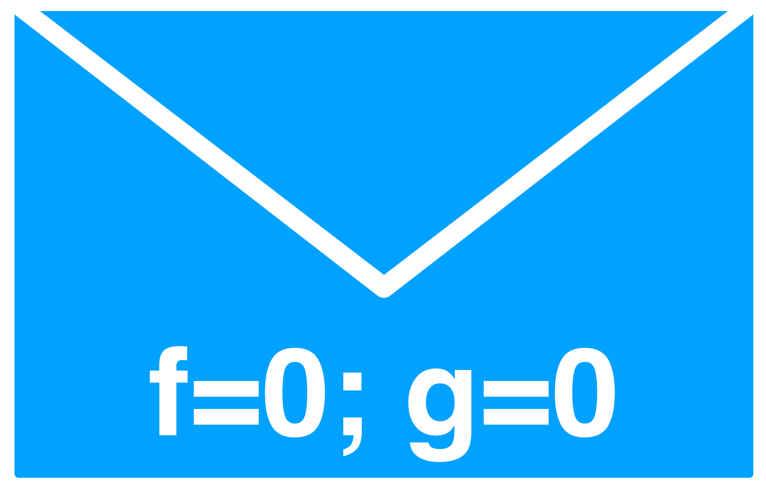
S is the set of states,
Pk is the set of all packets $\epsilon : S \times \mathsf{Pk} \to 2^{\mathsf{Pk}}$

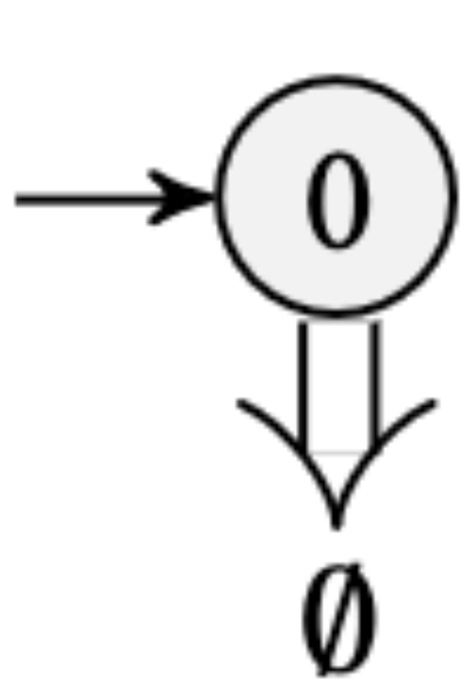
Checking Equivalence in NetKAT

Are these two NetKAT automata equivalent?

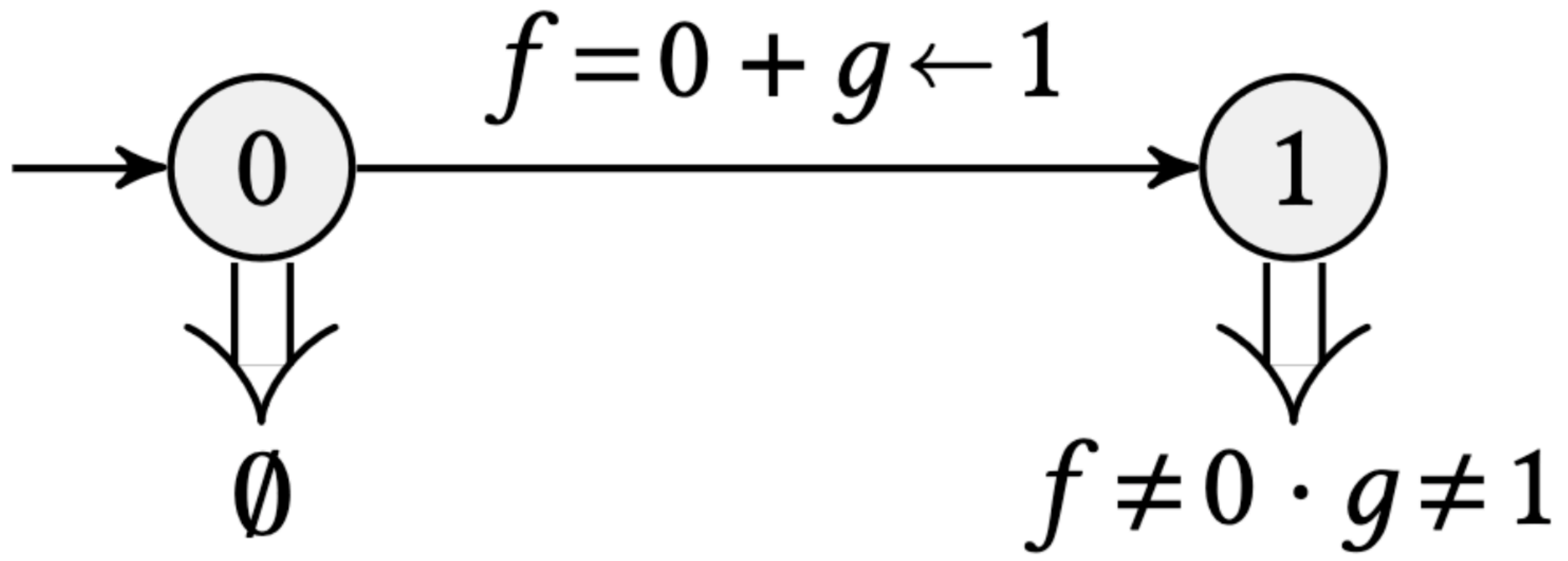
$\delta : S \times \mathsf{Pk} \to S^{\mathsf{Pk}}$







$$\begin{tikzcd} \arrow[r] & 0 \arrow[d, Rightarrow] \\ & \emptyset \end{tikzcd}$$



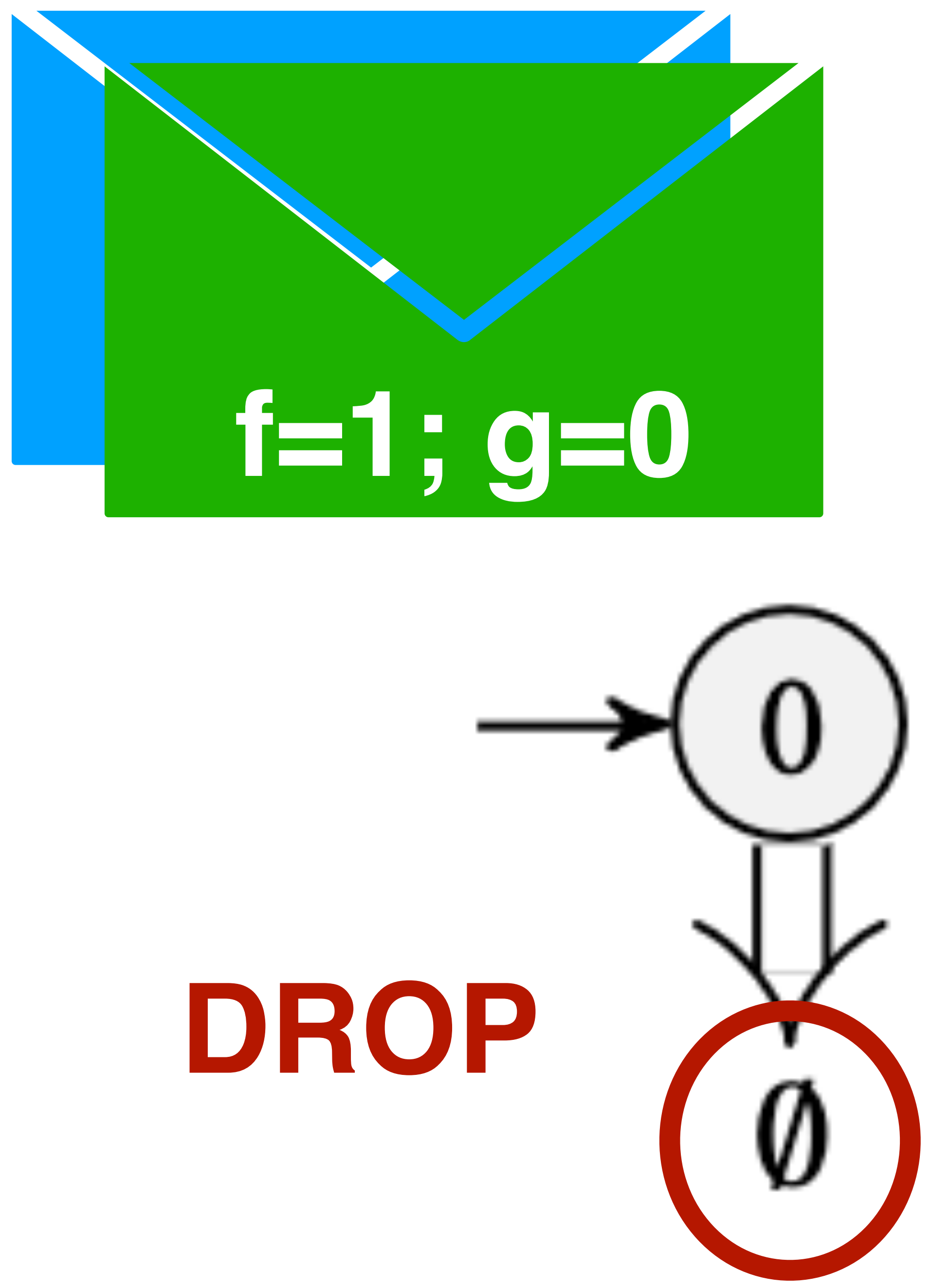
$$\begin{tikzcd} \arrow[r] & 0 \arrow[r, "f=0 + g\leftarrow 1"] \arrow[d, Rightarrow] & 1 \arrow[d, Rightarrow] \\ & \emptyset & f\neq 0 \cdot g\neq 1 \end{tikzcd}$$

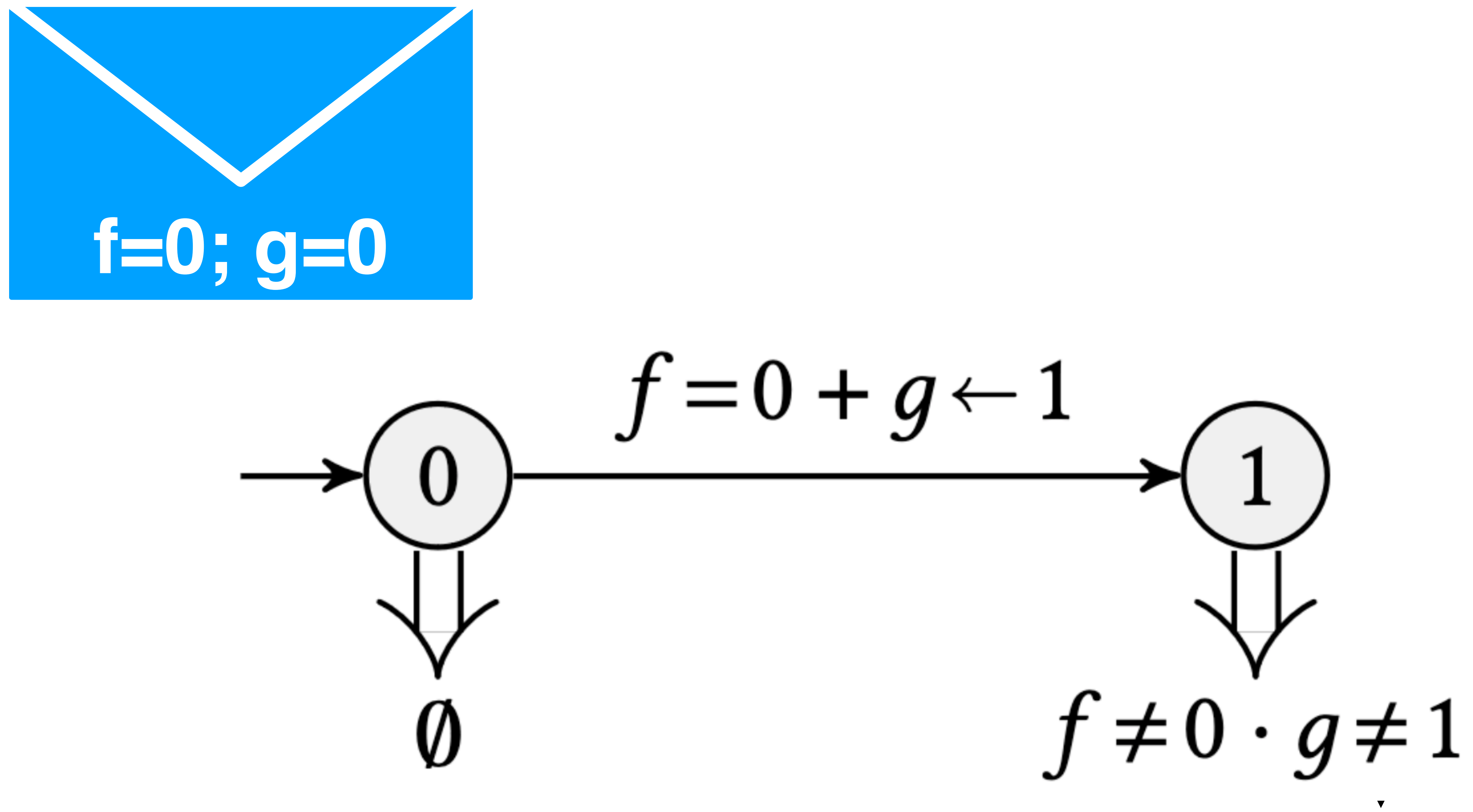
S is the set of states,
Pk is the set of all packets $\epsilon : S \times \mathsf{Pk} \to 2^{\mathsf{Pk}}$

Checking Equivalence in NetKAT

Are these two NetKAT automata equivalent?

$$\delta : S \times \mathsf{Pk} \to S^{\mathsf{Pk}}$$



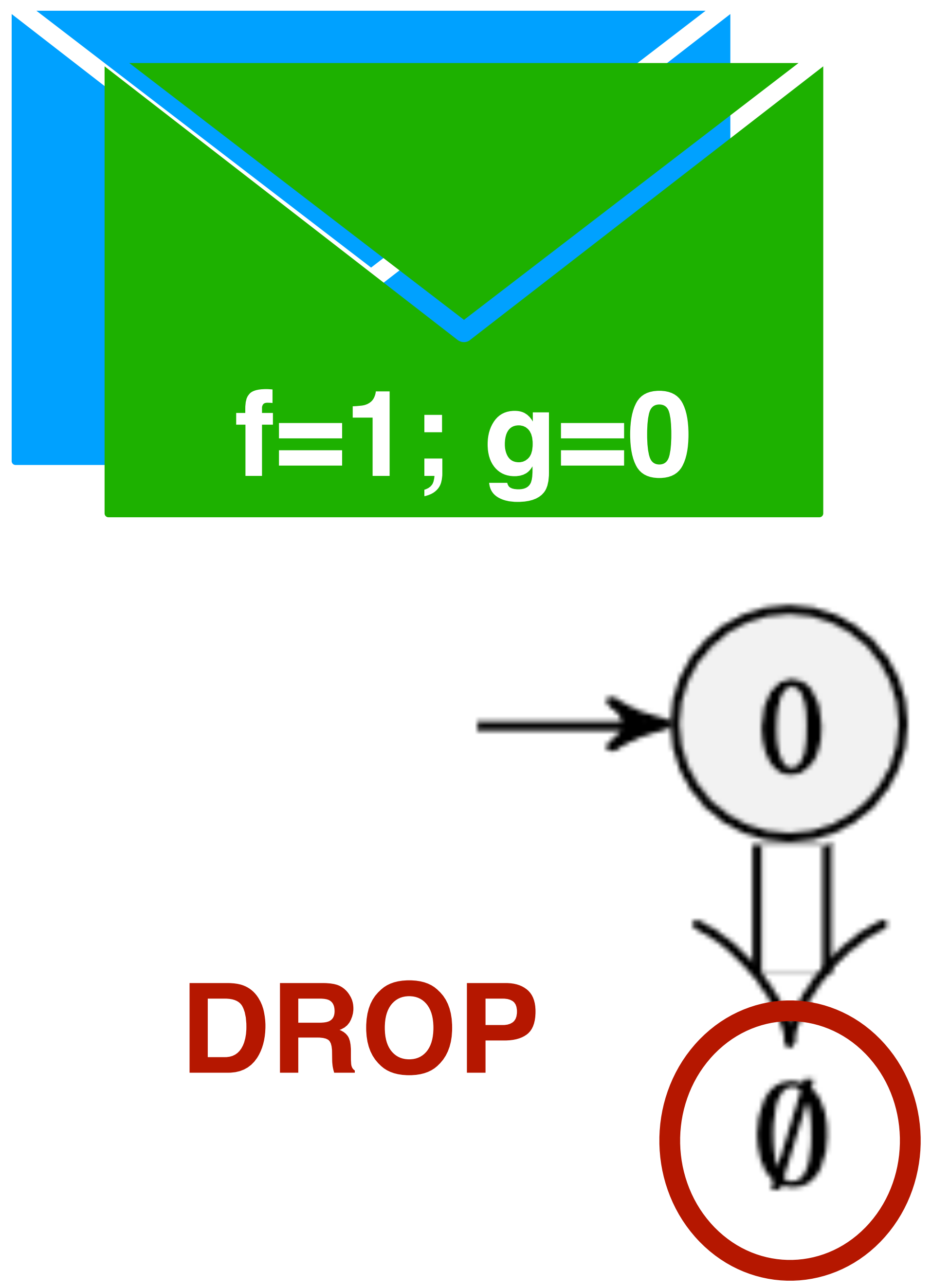


S is the set of states,
Pk is the set of all packets $\epsilon : S \times \mathsf{Pk} \to 2^{\mathsf{Pk}}$

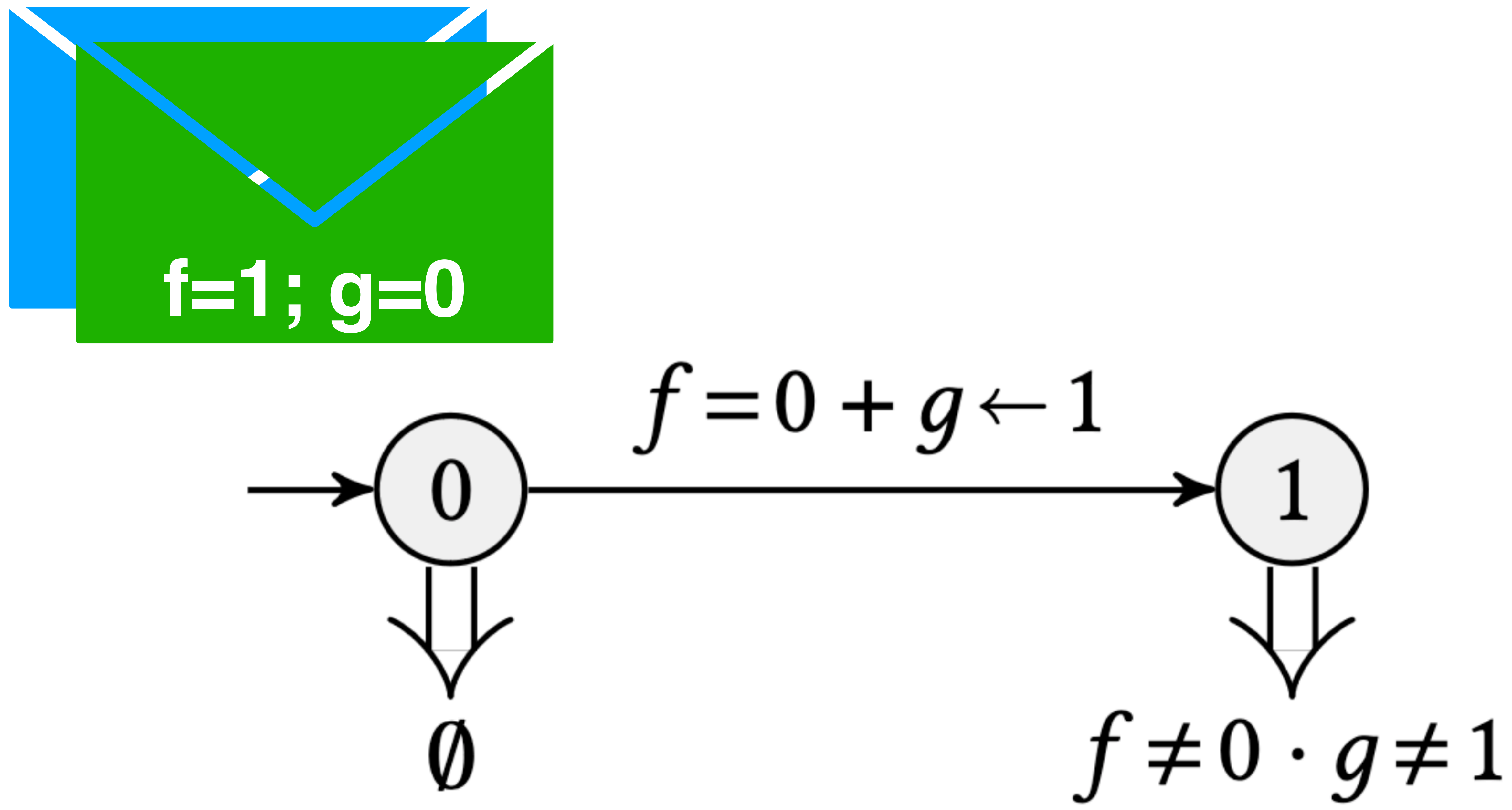
Checking Equivalence in NetKAT

Are these two NetKAT automata equivalent?

$\delta : S \times \mathsf{Pk} \to S^{\mathsf{Pk}}$



$$\begin{tikzcd} \arrow[r] & 0 \arrow[d, Rightarrow] \\ & \emptyset \end{tikzcd}$$



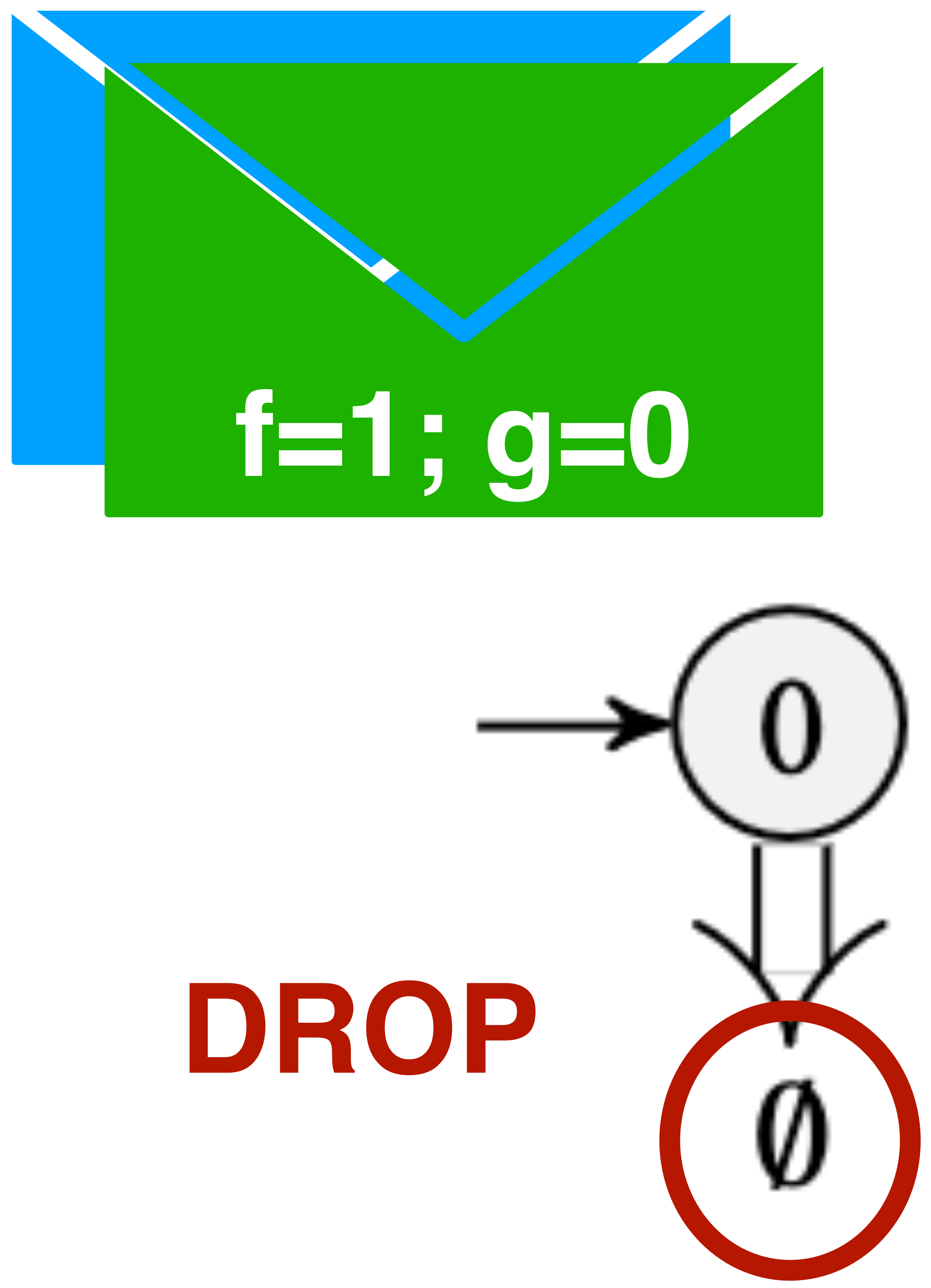
$$\begin{tikzcd} \arrow[r] & 0 \arrow[r, "f=0+g\leftarrow 1"] \arrow[d, Rightarrow] & 1 \arrow[d, Rightarrow] \\ & \emptyset & f\neq 0 \cdot g\neq 1 \end{tikzcd}$$

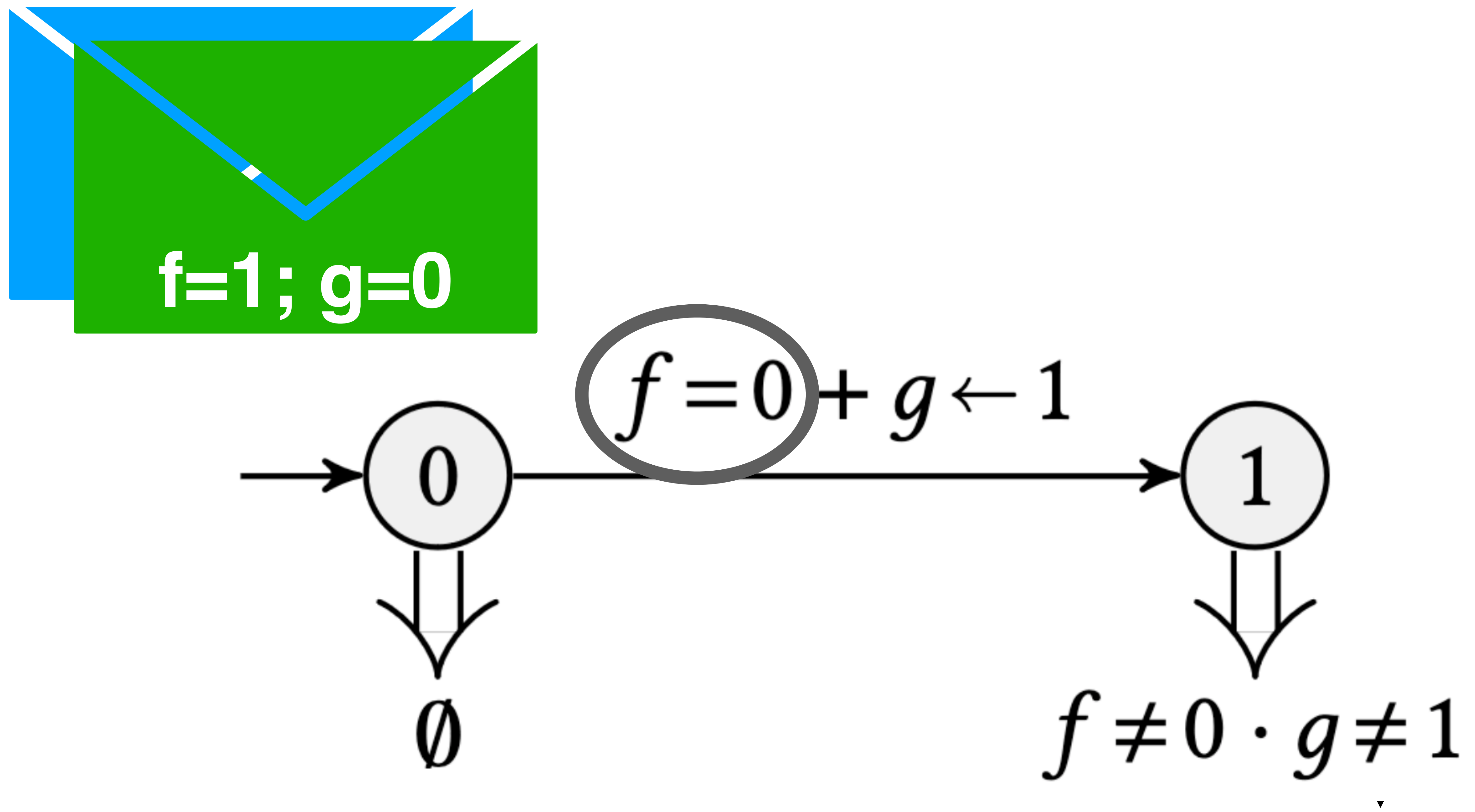
S is the set of states,
Pk is the set of all packets $\epsilon : S \times \mathsf{Pk} \to 2^{\mathsf{Pk}}$

Checking Equivalence in NetKAT

Are these two NetKAT automata equivalent?

$\delta : S \times \mathsf{Pk} \to S^{\mathsf{Pk}}$





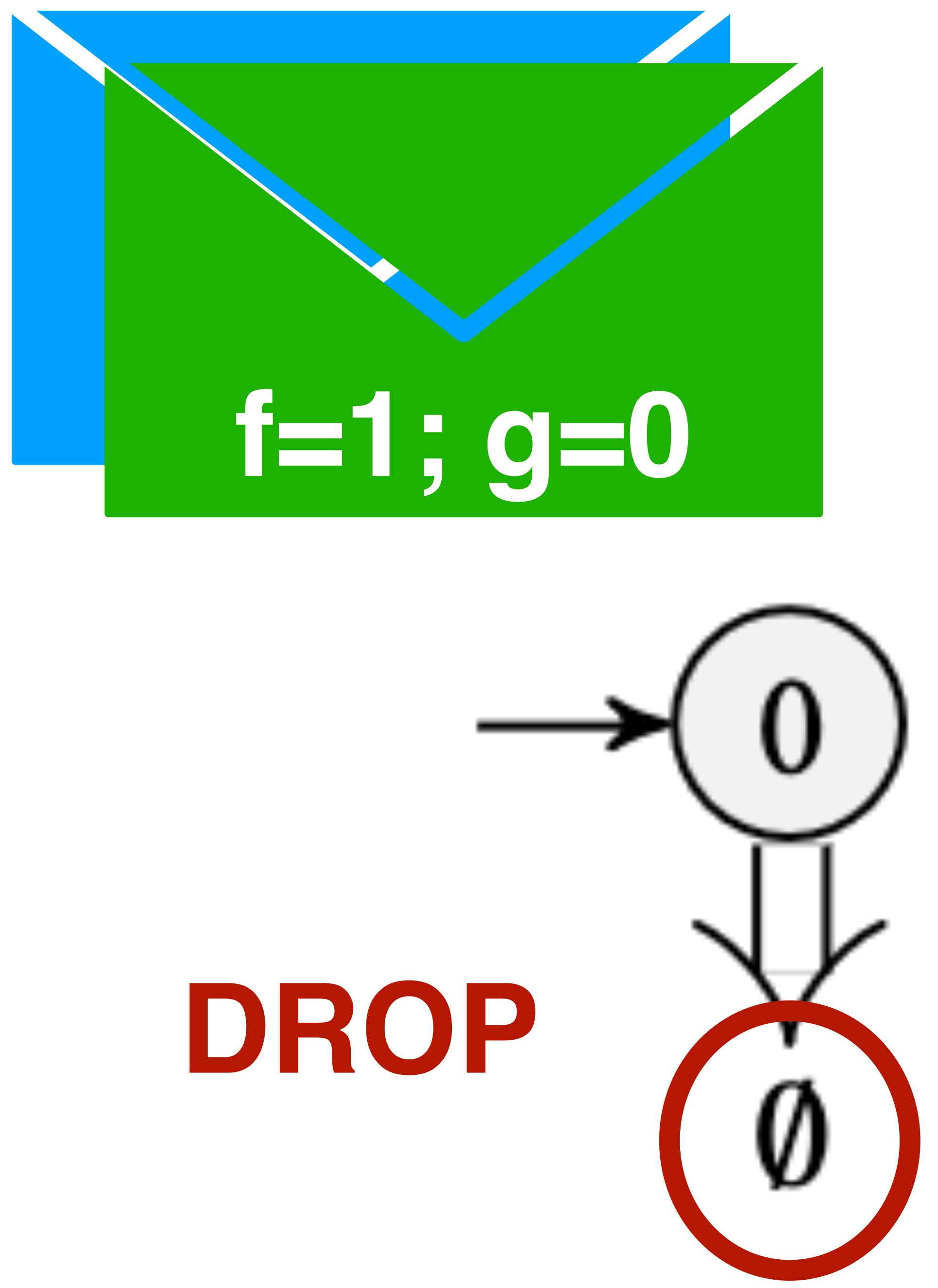
S is the set of states,
Pk is the set of all packets

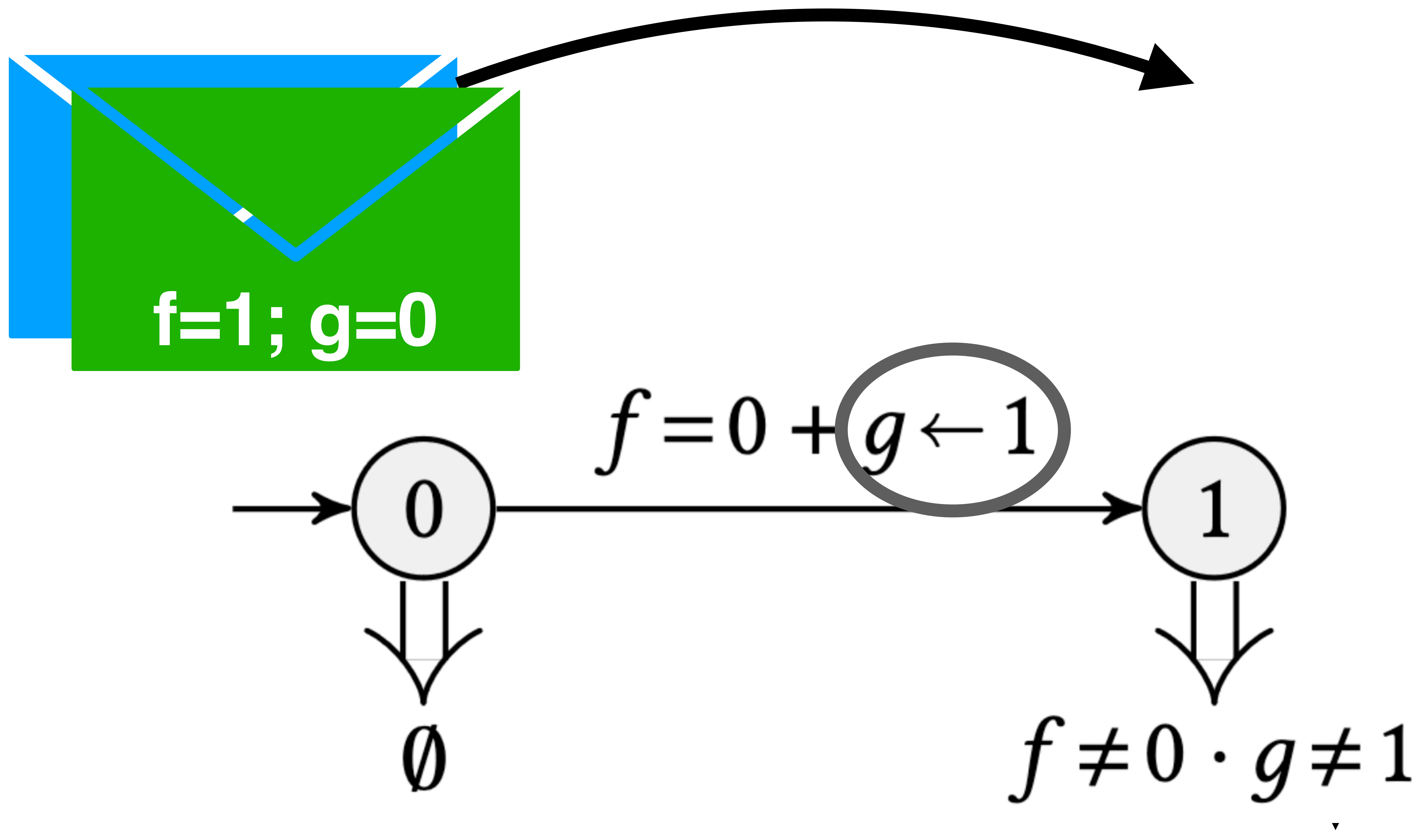
$\epsilon : S \times \mathsf{Pk} \to 2^{\mathsf{Pk}}$

Checking Equivalence in NetKAT

Are these two NetKAT automata equivalent?

$\delta : S \times \mathsf{Pk} \to S^{\mathsf{Pk}}$



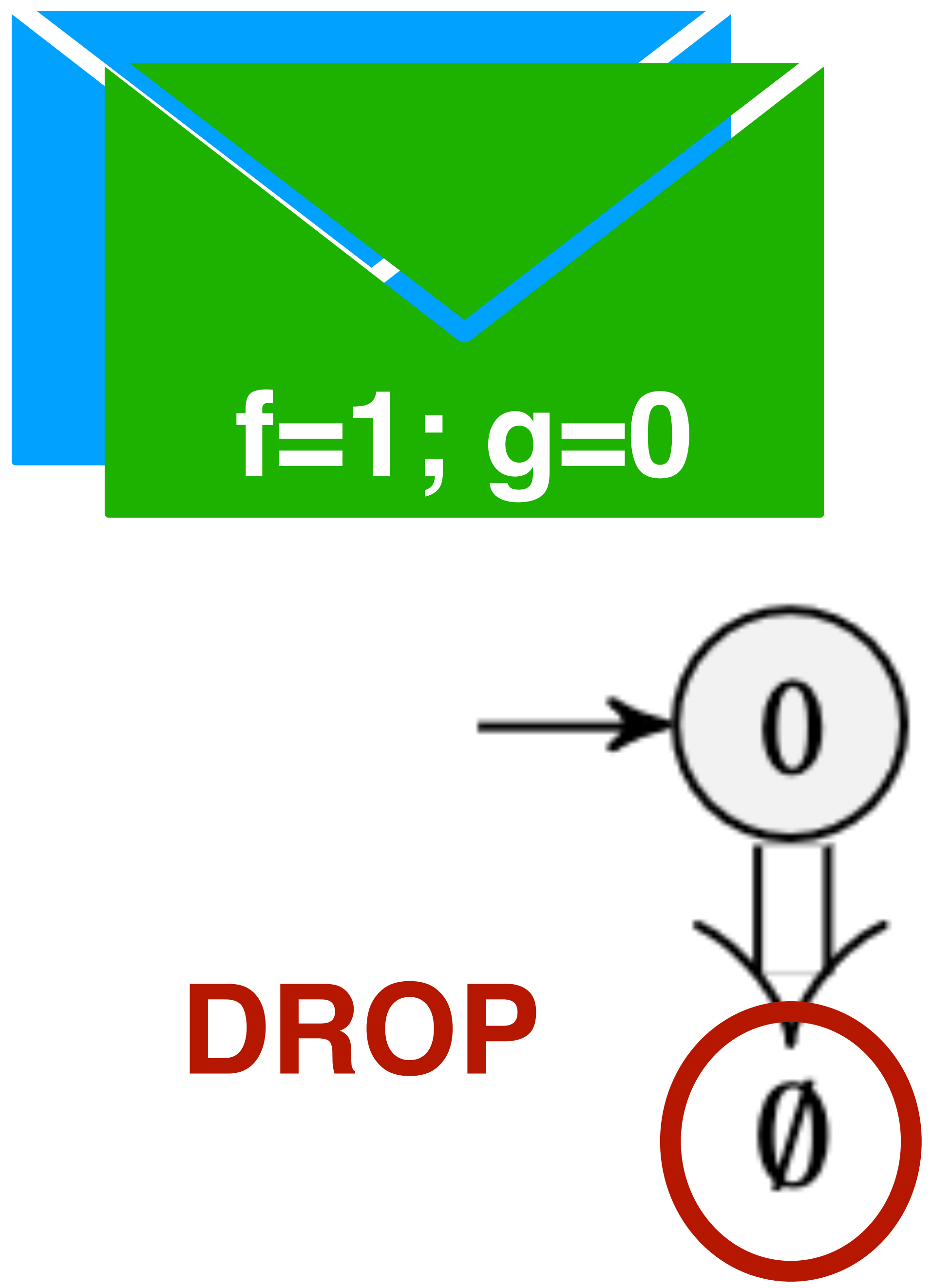


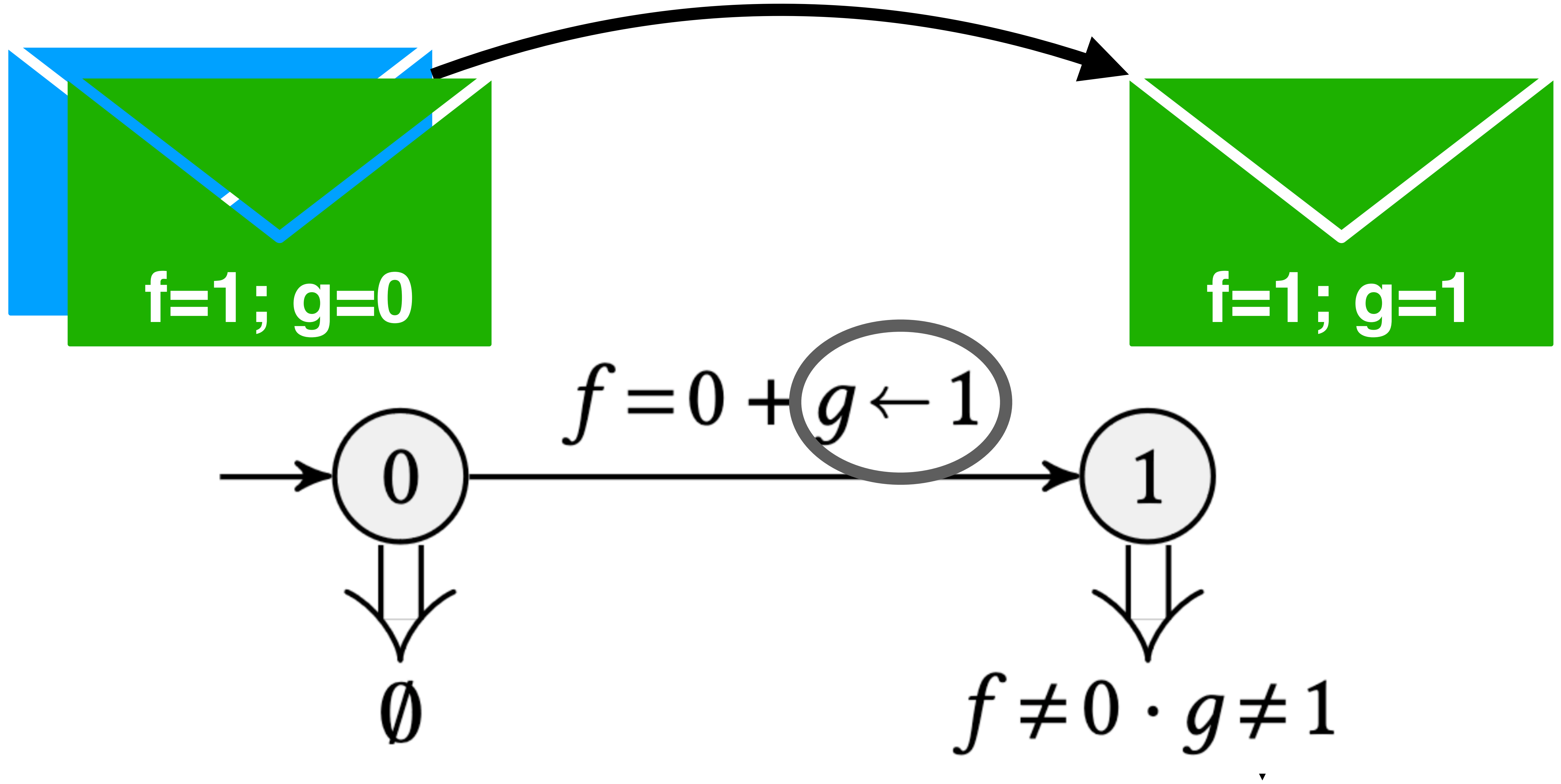
S is the set of states,
Pk is the set of all packets $\epsilon : S \times \mathsf{Pk} \to 2^{\mathsf{Pk}}$

Checking Equivalence in NetKAT

Are these two NetKAT automata equivalent?

$\delta : S \times \mathsf{Pk} \to S^{\mathsf{Pk}}$



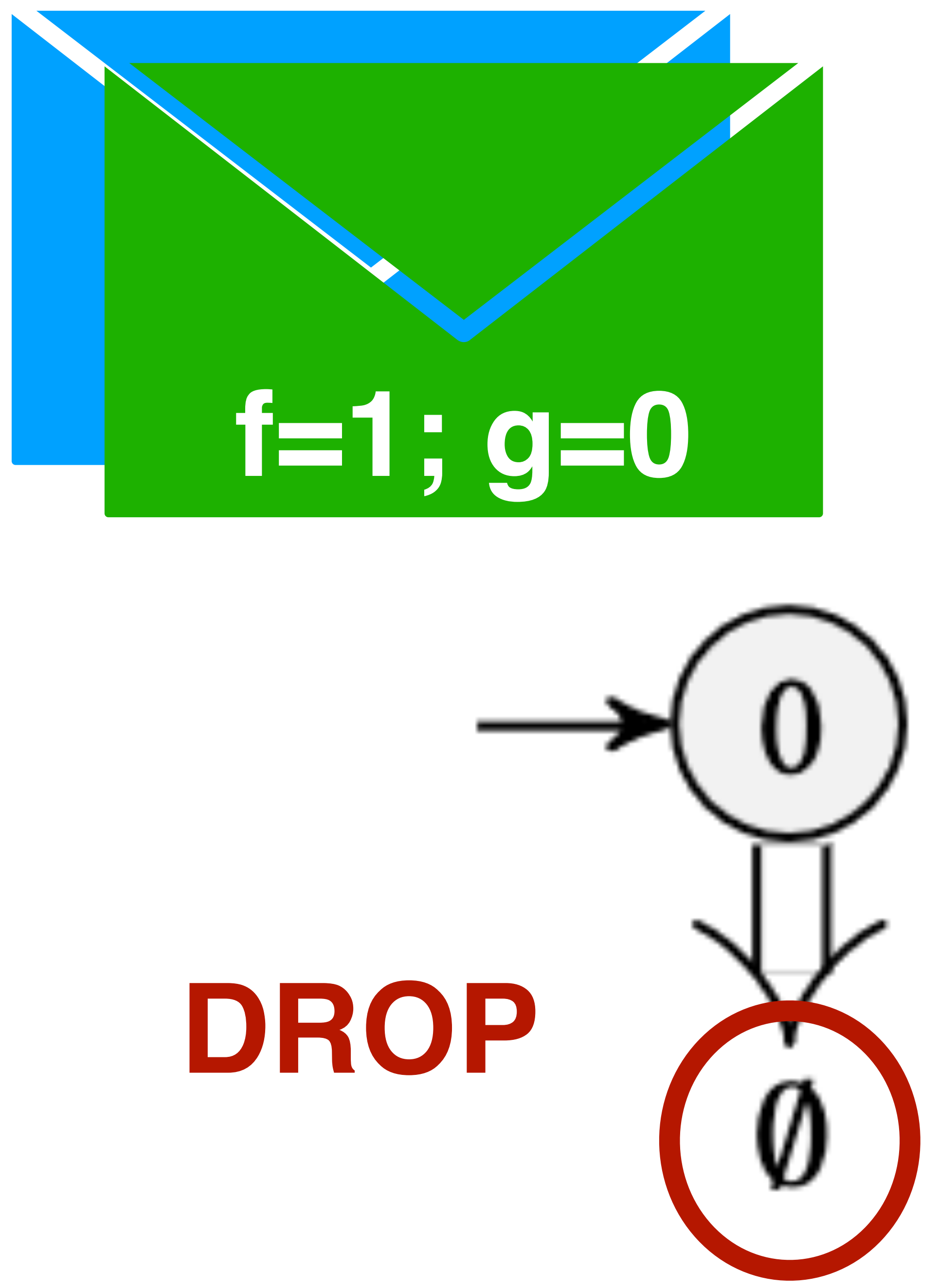


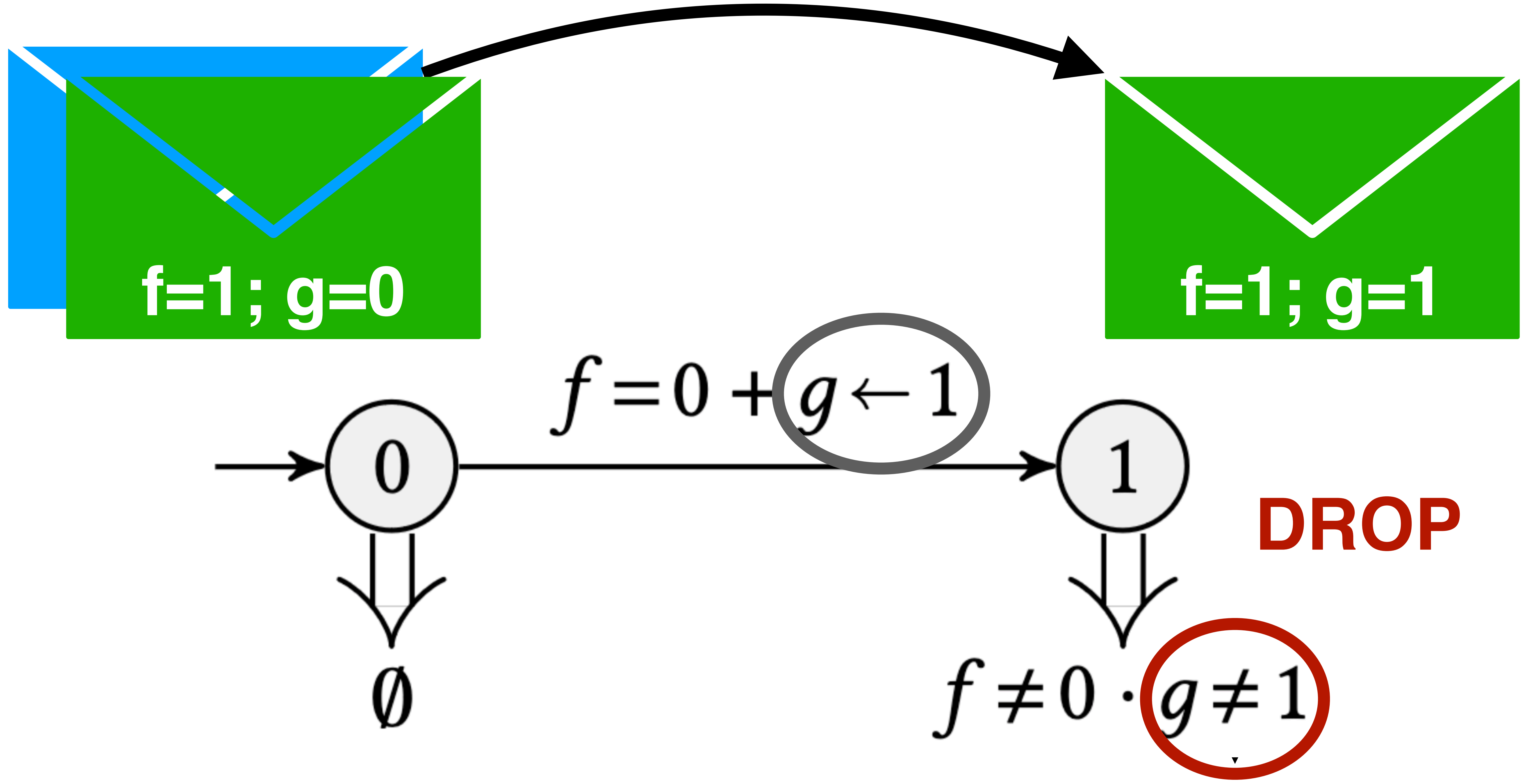
S is the set of states,
Pk is the set of all packets $\epsilon : S \times \mathsf{Pk} \to 2^{\mathsf{Pk}}$

Checking Equivalence in NetKAT

Are these two NetKAT automata equivalent?

$\delta : S \times \mathsf{Pk} \rightarrow S^{\mathsf{Pk}}$





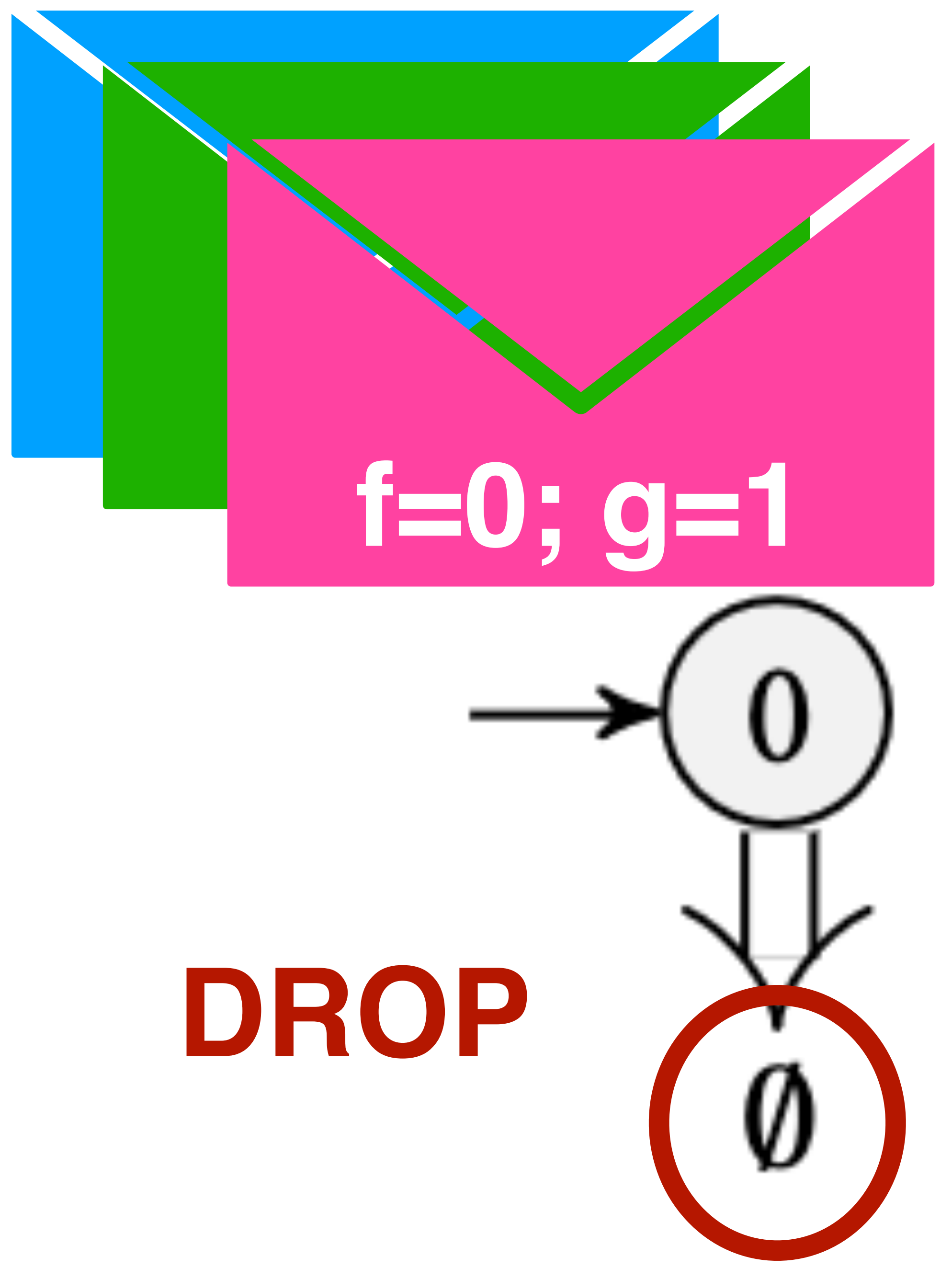
S is the set of states,
Pk is the set of all packets

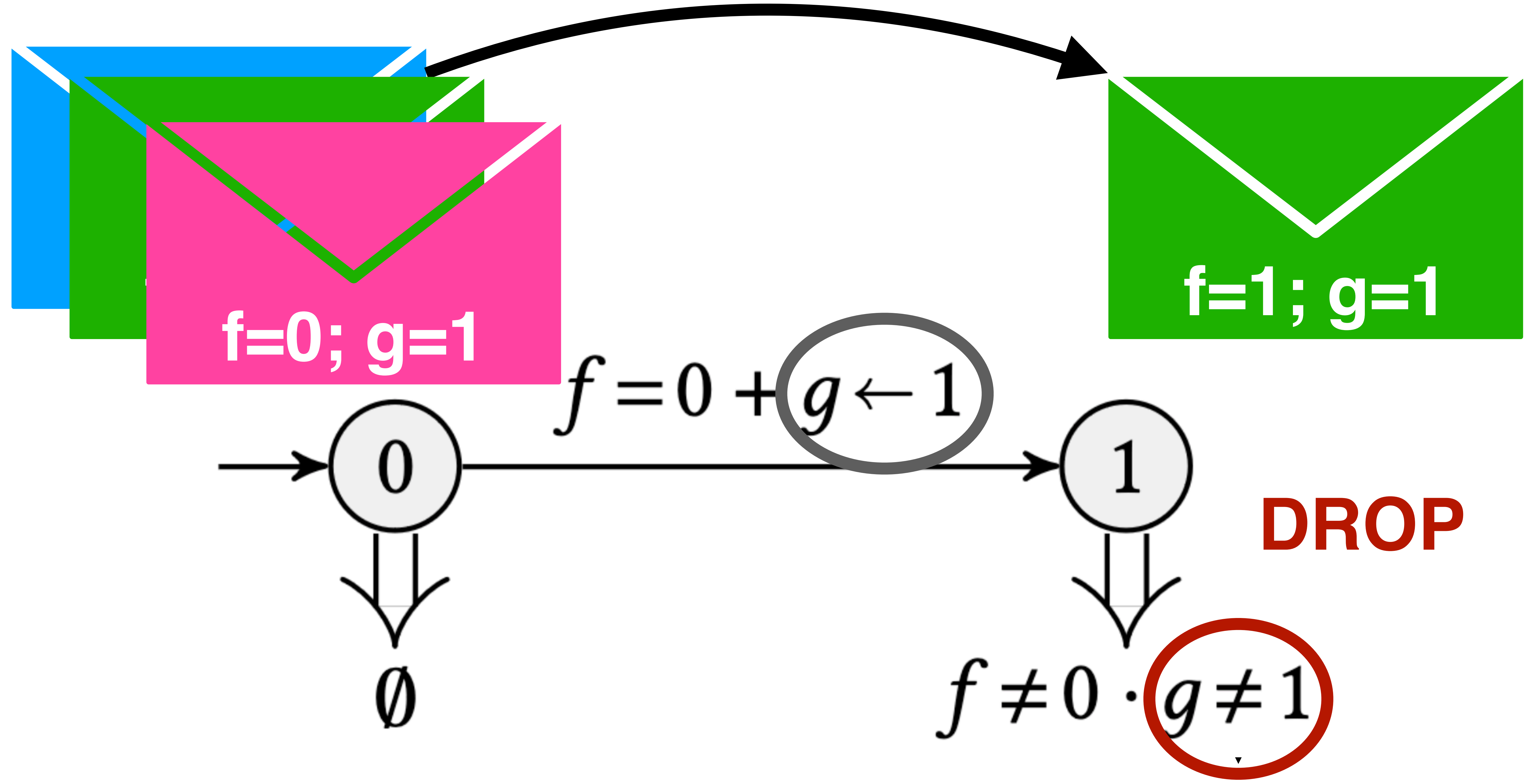
$\epsilon : S \times \mathsf{Pk} \rightarrow 2^{\mathsf{Pk}}$

Checking Equivalence in NetKAT

Are these two NetKAT automata equivalent?

$\delta : S \times \text{Pk} \to S^{\text{Pk}}$



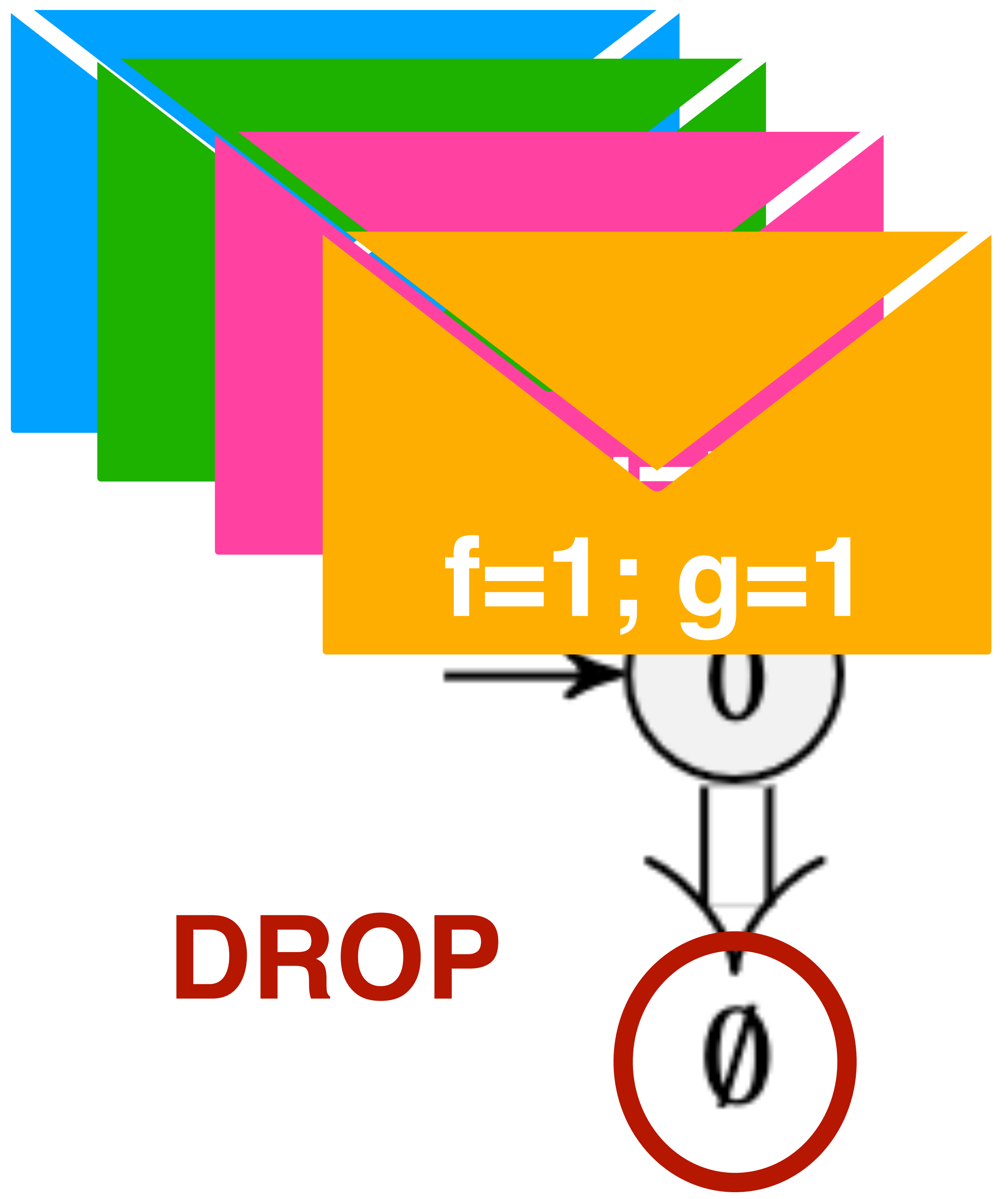


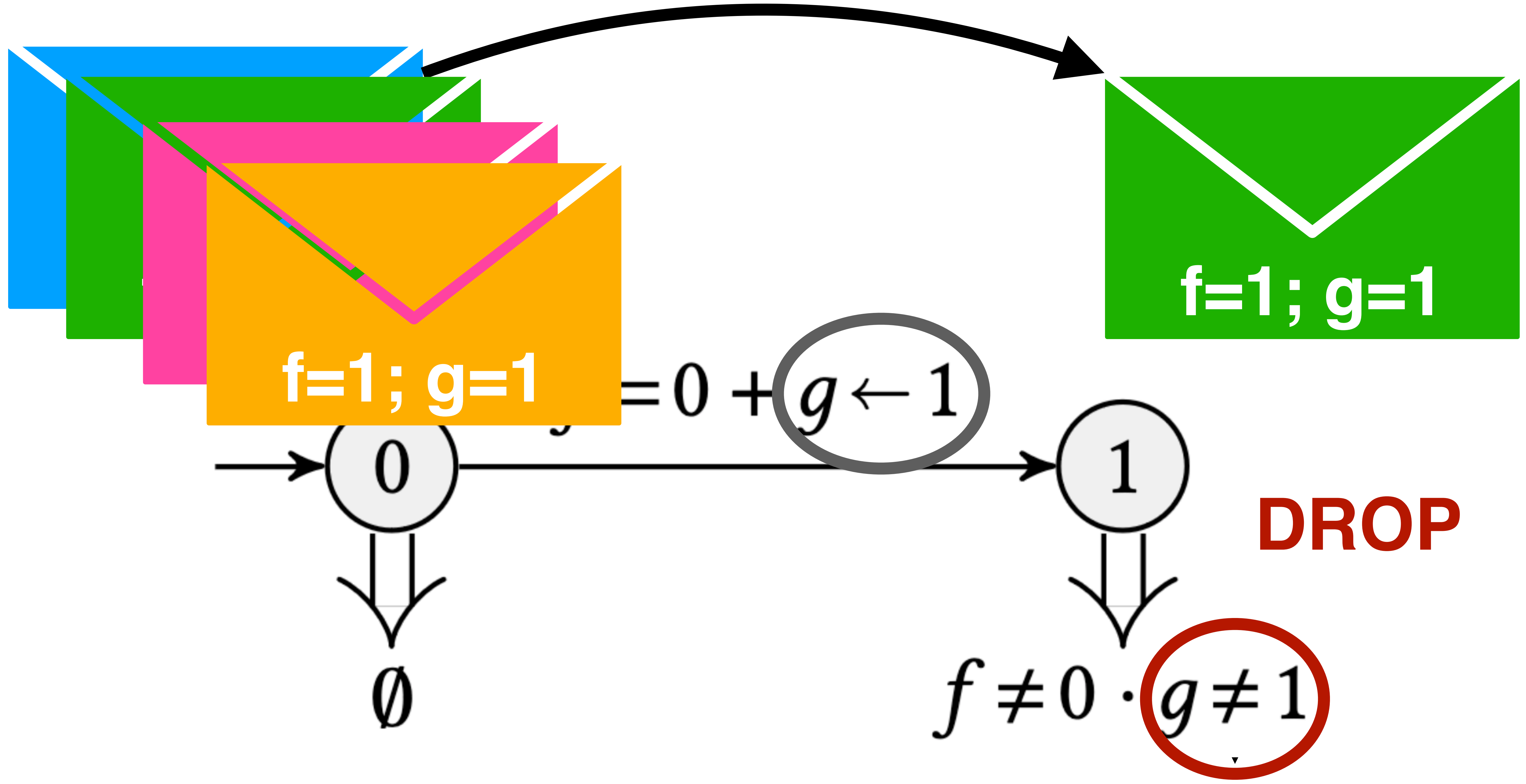
S is the set of states,
Pk is the set of all packets $\epsilon : S \times \text{Pk} \to 2^{\text{Pk}}$

Checking Equivalence in NetKAT

Are these two NetKAT automata equivalent?

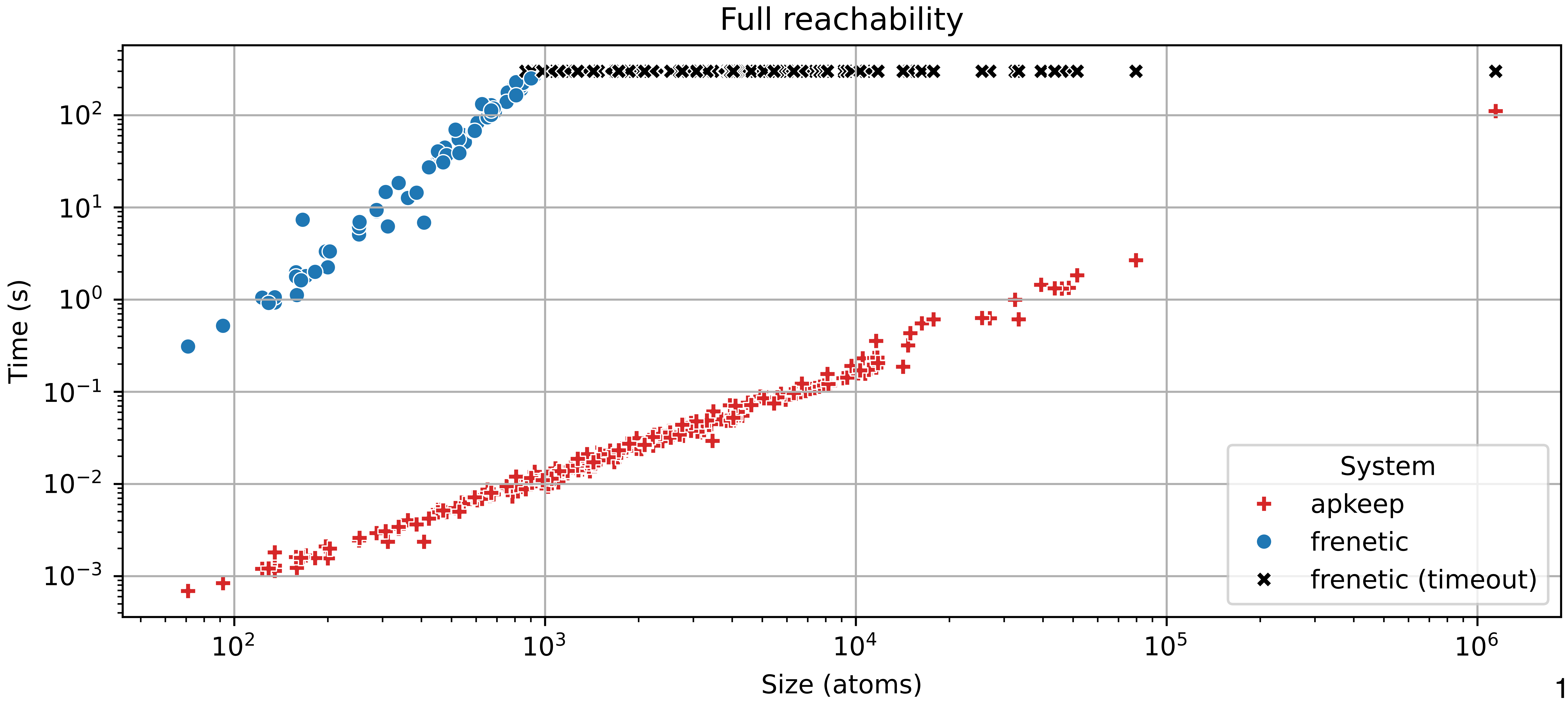
$\delta : S \times \text{Pk} \to S^{\text{Pk}}$





S is the set of states,
Pk is the set of all packets $\epsilon : S \times \text{Pk} \to 2^{\text{Pk}}$

NetKAT and APKeep (NSDI 2020)



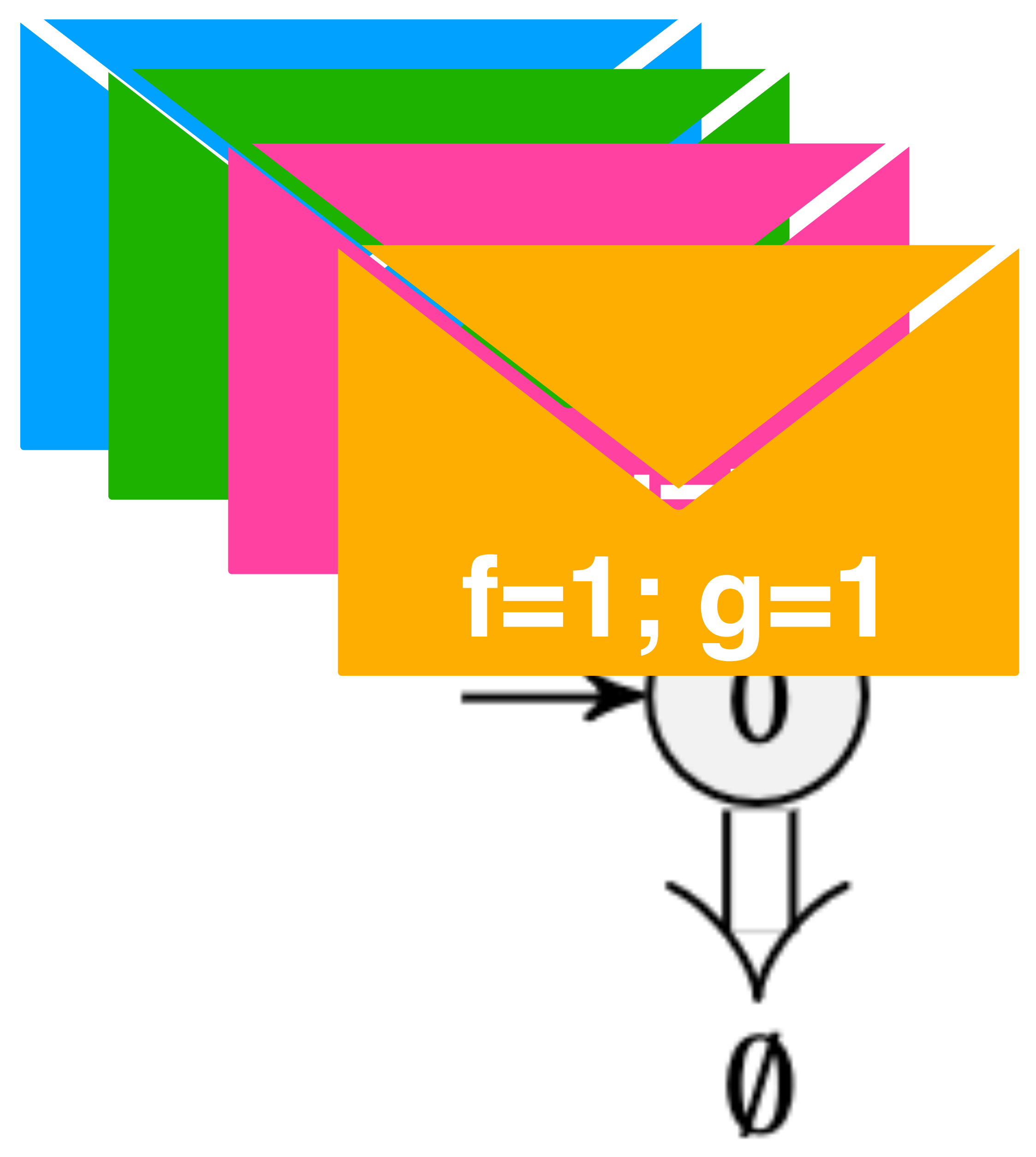
Problem:
NetKAT is limited in practice

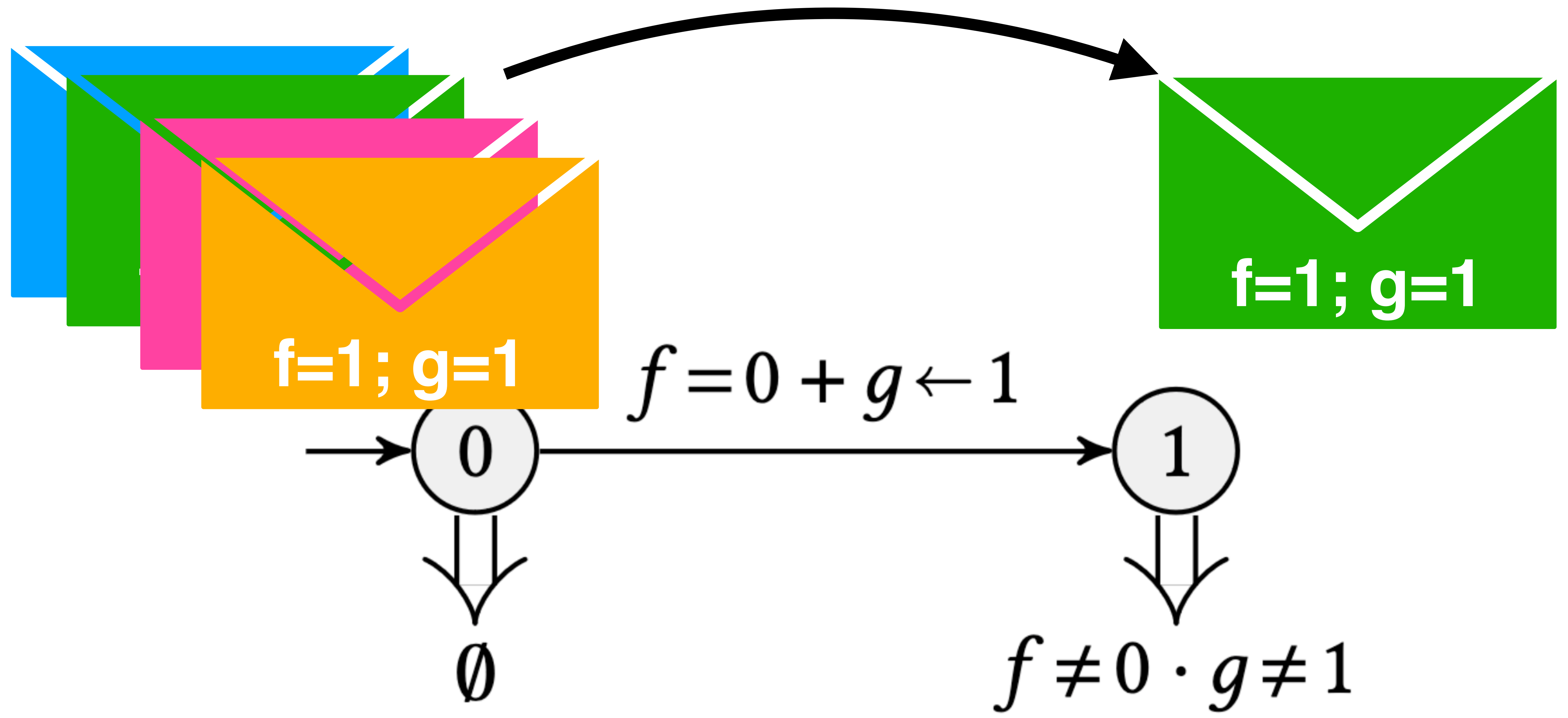
Problem:
NetKAT is limited in practice

Contributions (This work, PLDI 2024):

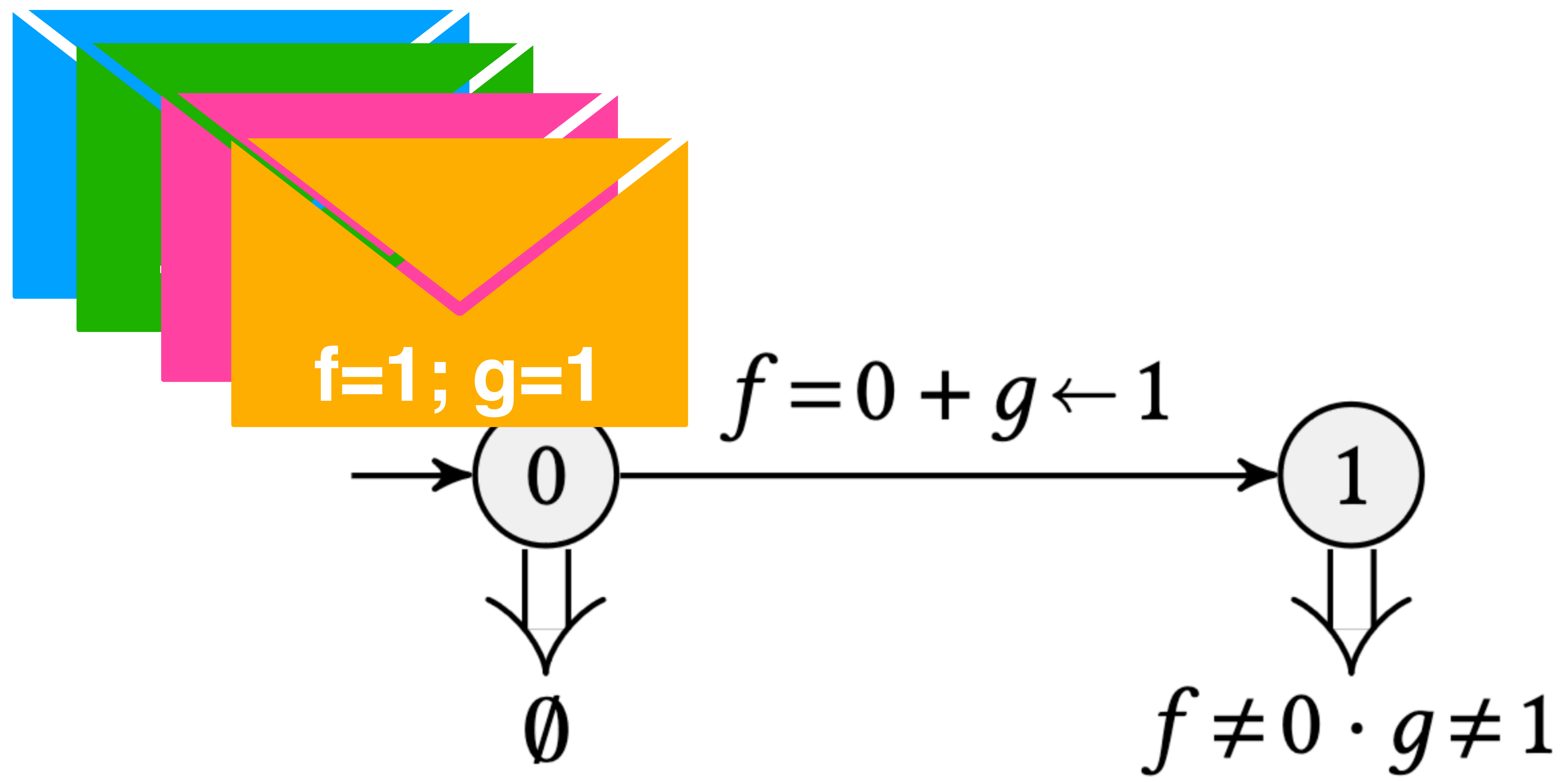
1. Symbolic packets and techniques

Represent Sets of Packets Symbolically



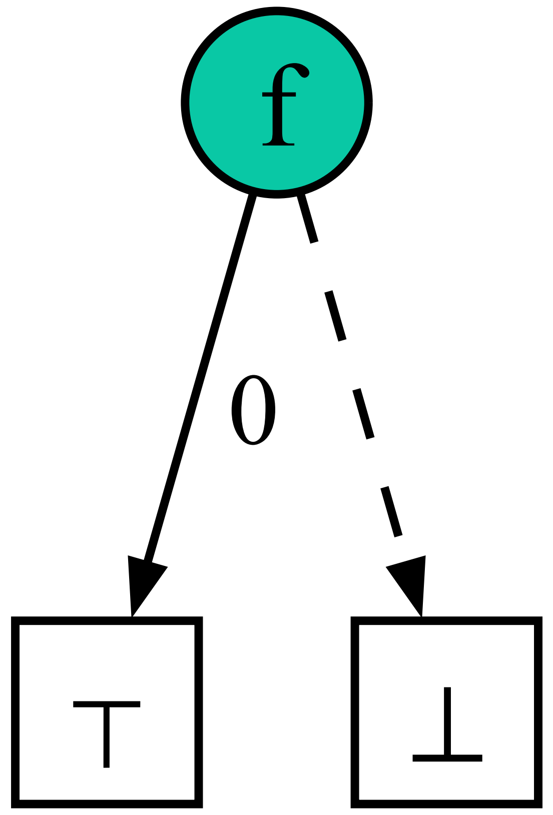


Represent Sets of Packets Symbolically

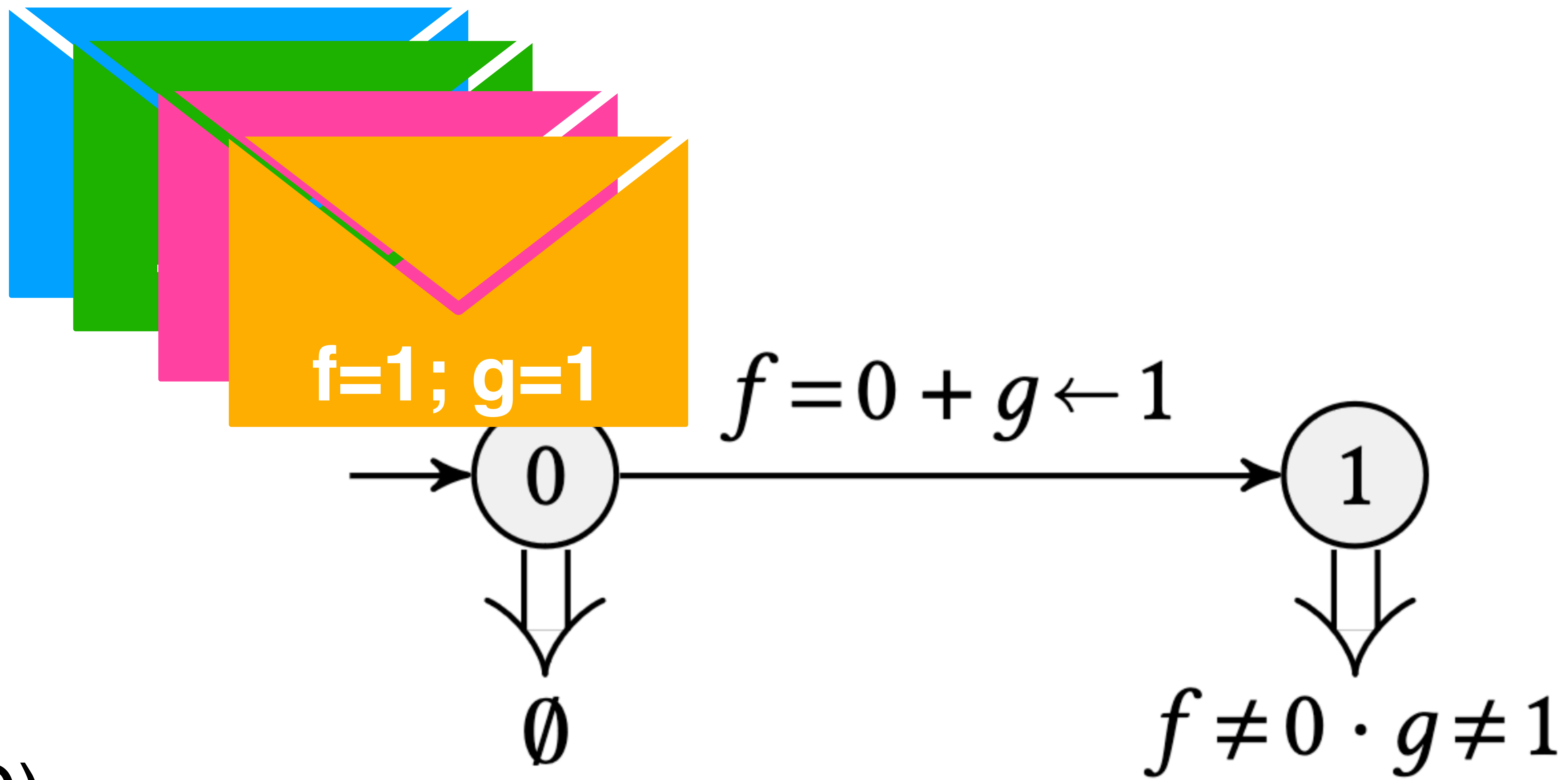


Represent Sets of Packets Symbolically

$$\{\alpha \in \mathsf{Pk} \mid \alpha.f = 0\}$$



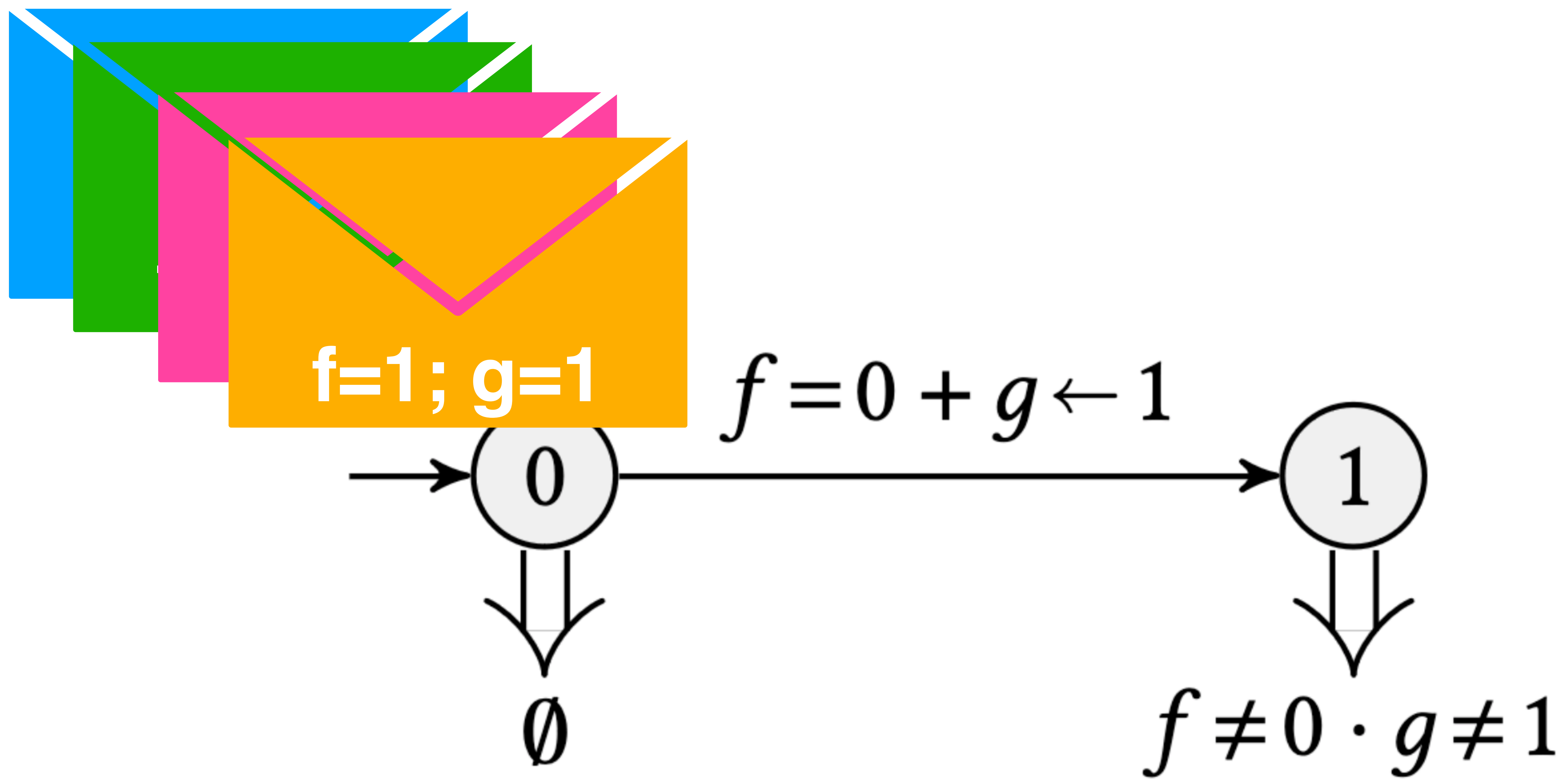
Binary Decision Diagram (BDD)



Represent Sets of Packets Symbolically

Set of all packets

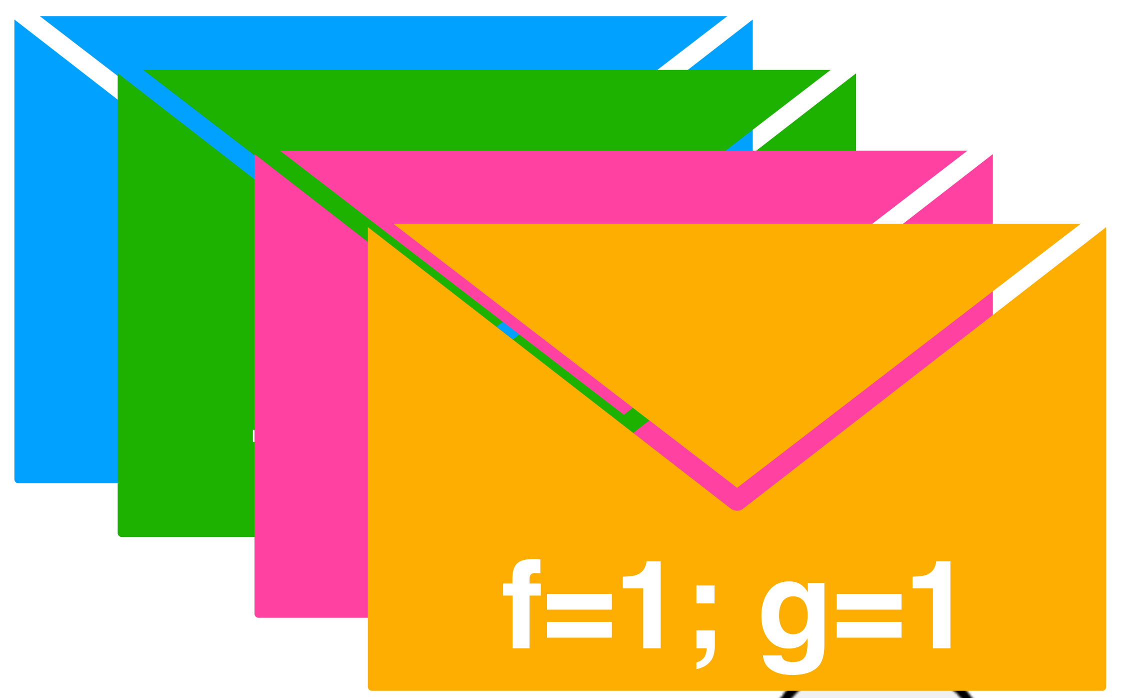
Pk

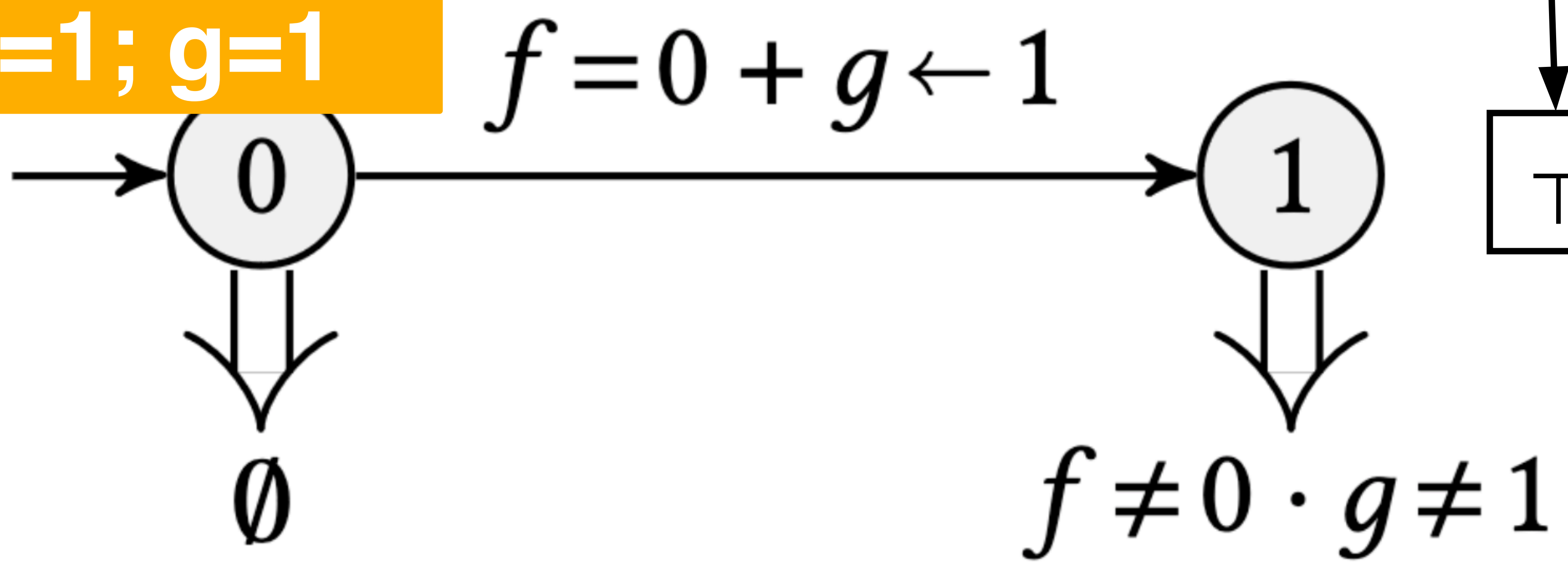


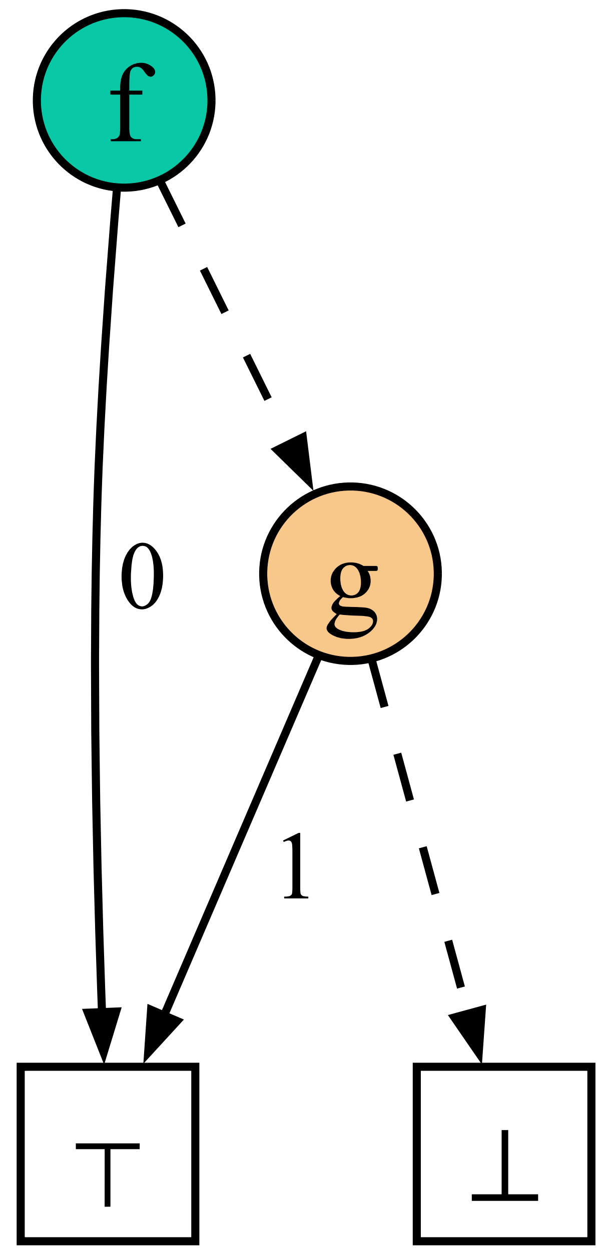
Represent Sets of Packets Symbolically

Set of all packets

Pk



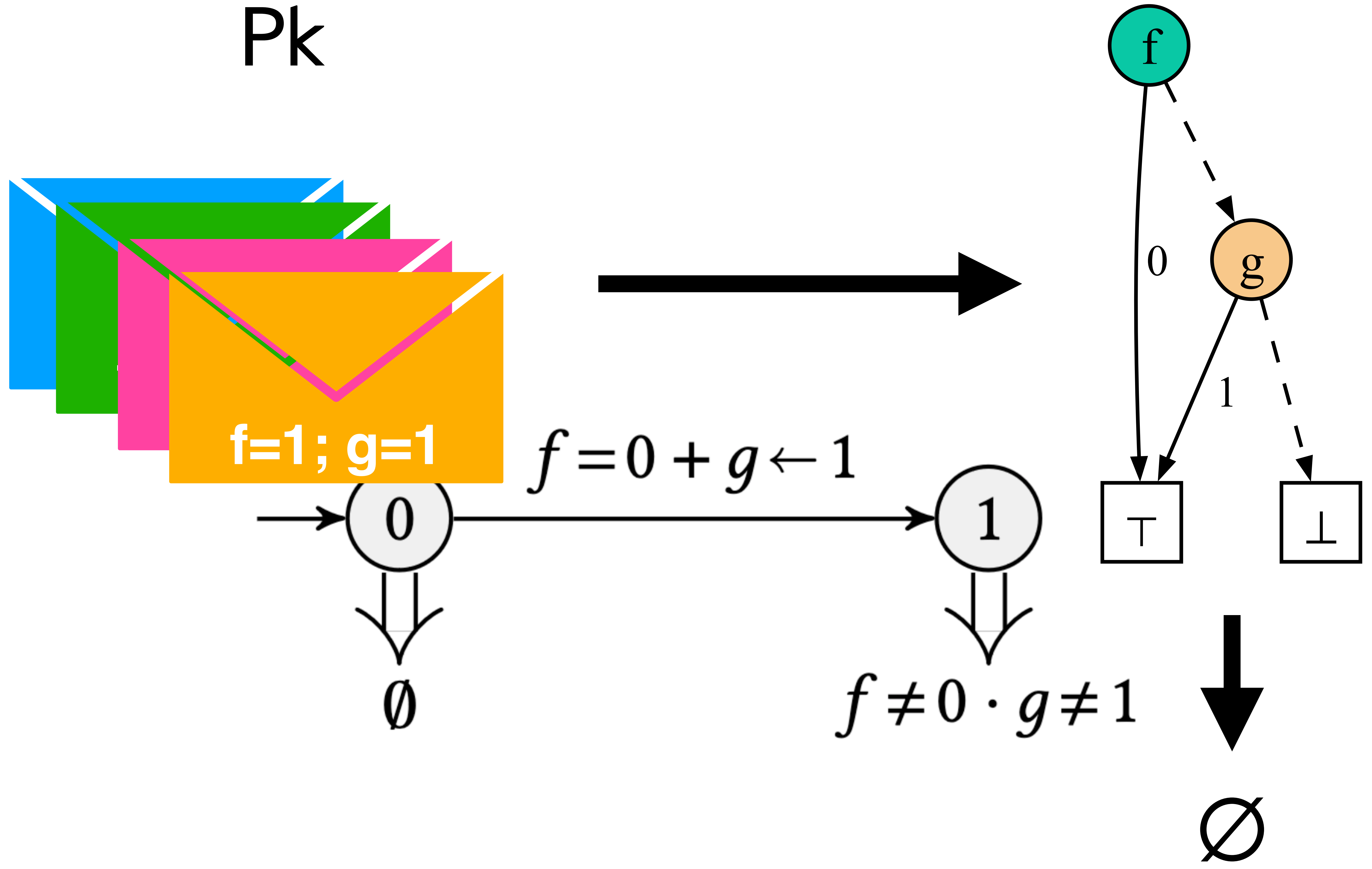




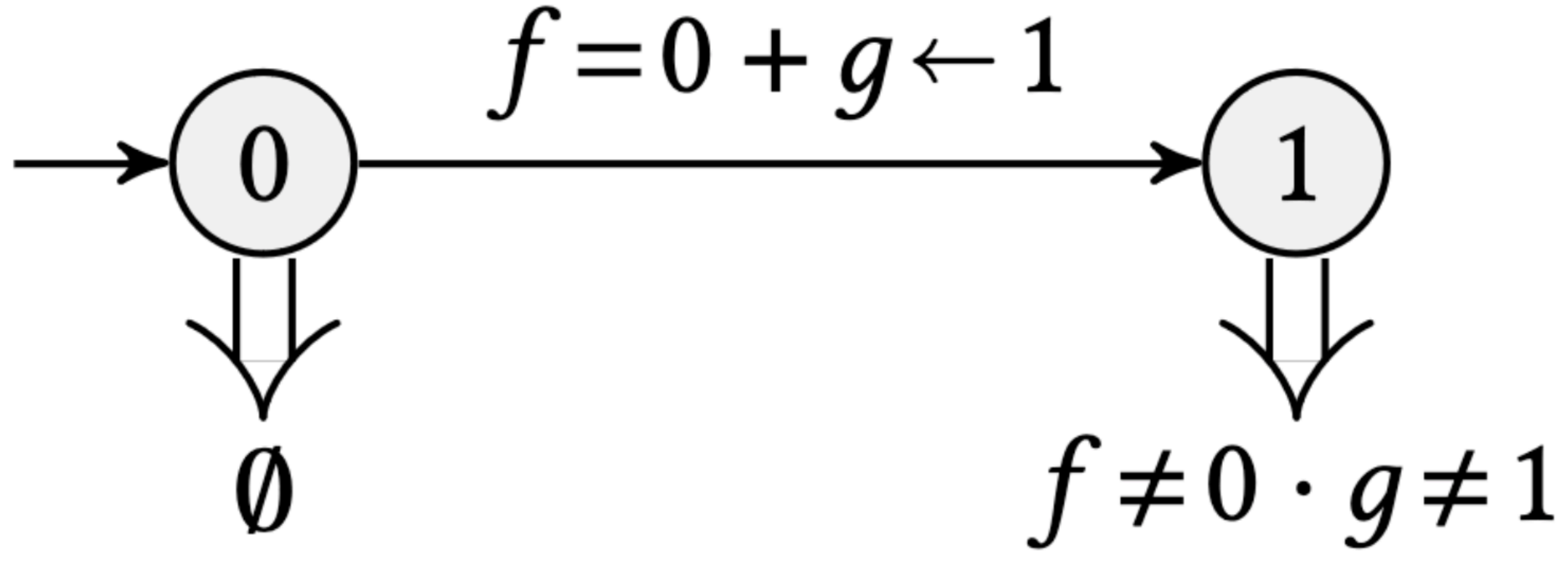
Represent Sets of Packets Symbolically

Set of all packets

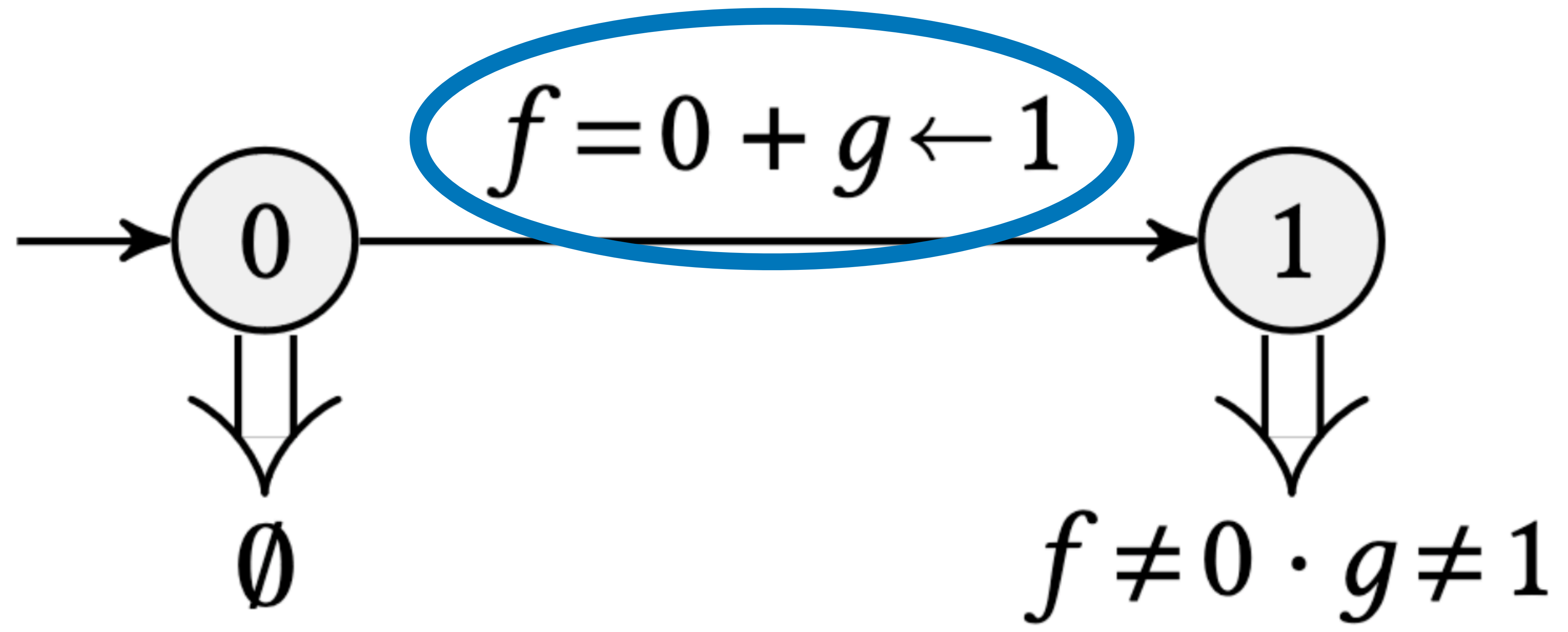
Pk



Represent Transitions Symbolically

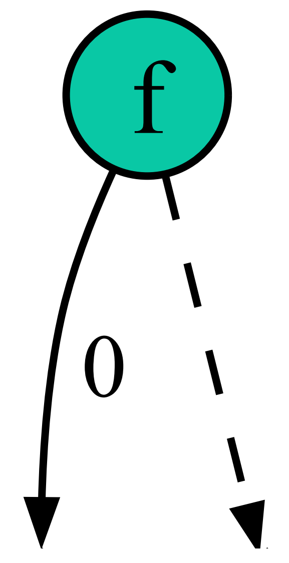


Represent Transitions Symbolically

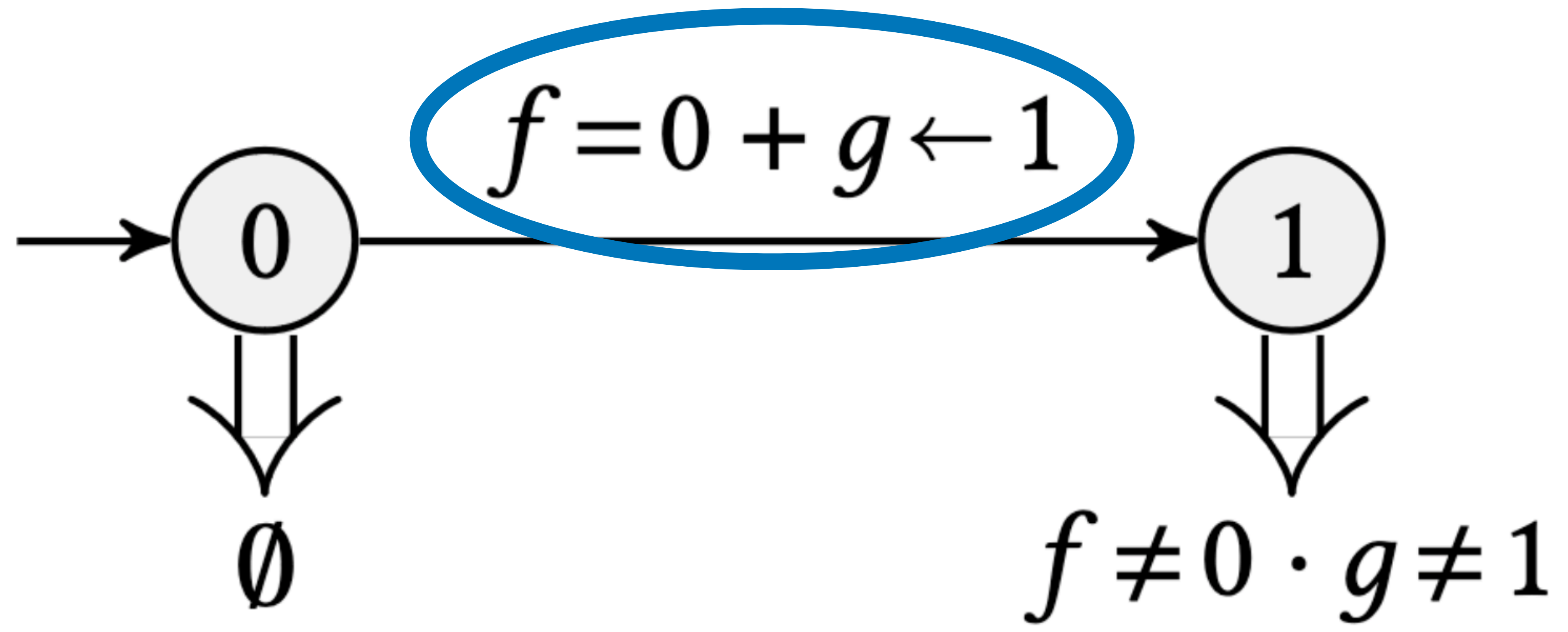


Represent Transitions Symbolically

Symbolic Packet Program (SPP)

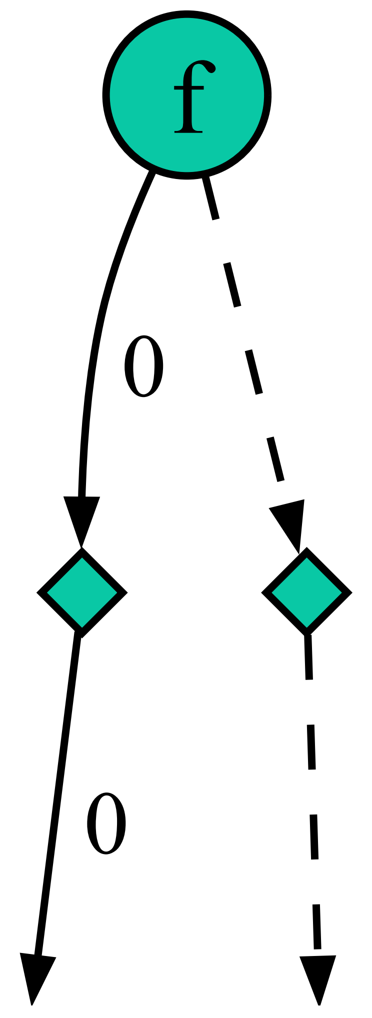


Circle-layers are filters



Represent Transitions Symbolically

Symbolic Packet Program (SPP)



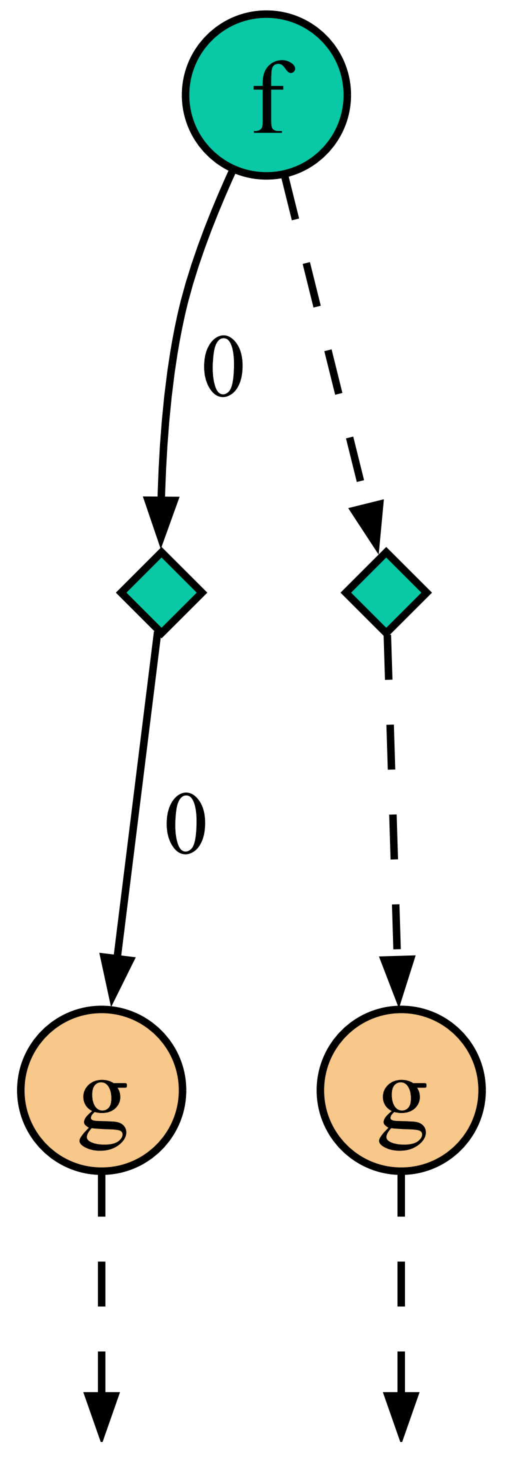
Circle-layers are filters

Diamond-layers are assignments

$$\begin{tikzcd} \arrow[r] & 0 \arrow[r, "f=0+g\leftarrow 1"] \arrow[d, Rightarrow] & 1 \arrow[d, Rightarrow] \\ & \emptyset & f\neq 0 \cdot g\neq 1 \end{tikzcd}$$

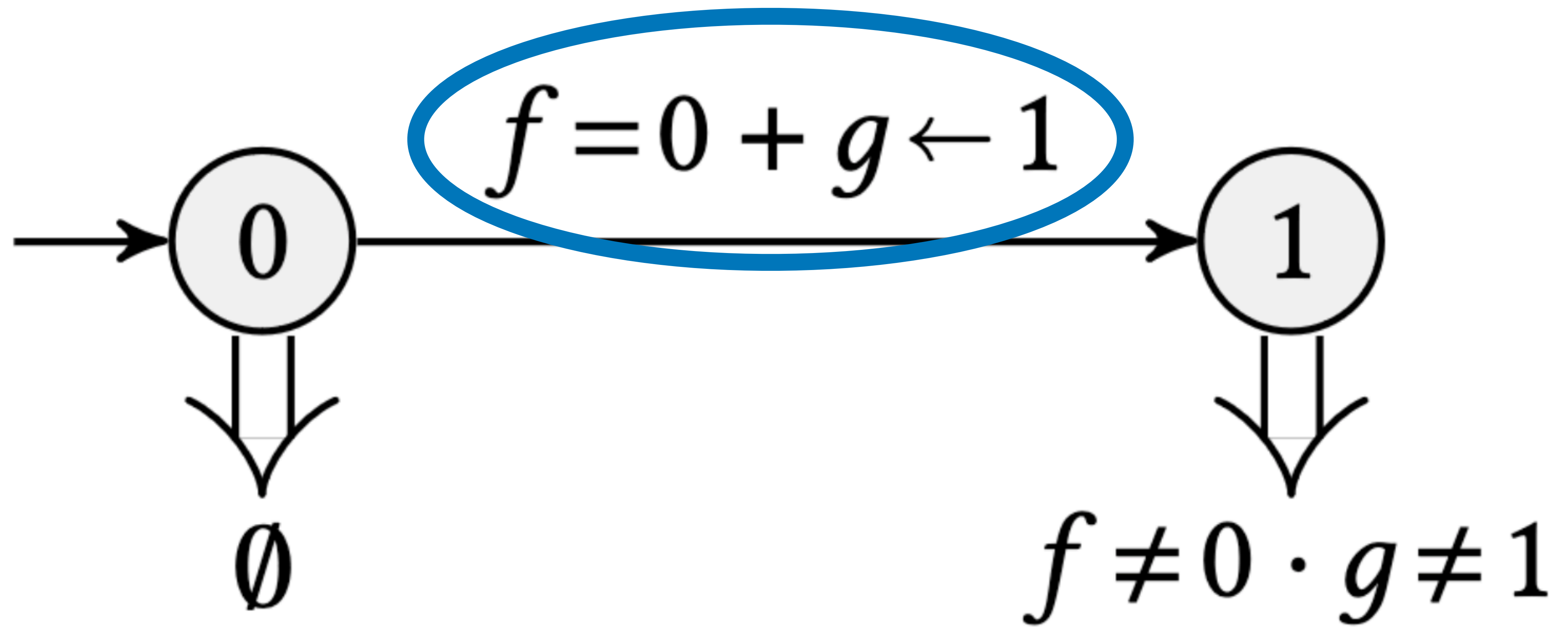
Represent Transitions Symbolically

Symbolic Packet Program (SPP)



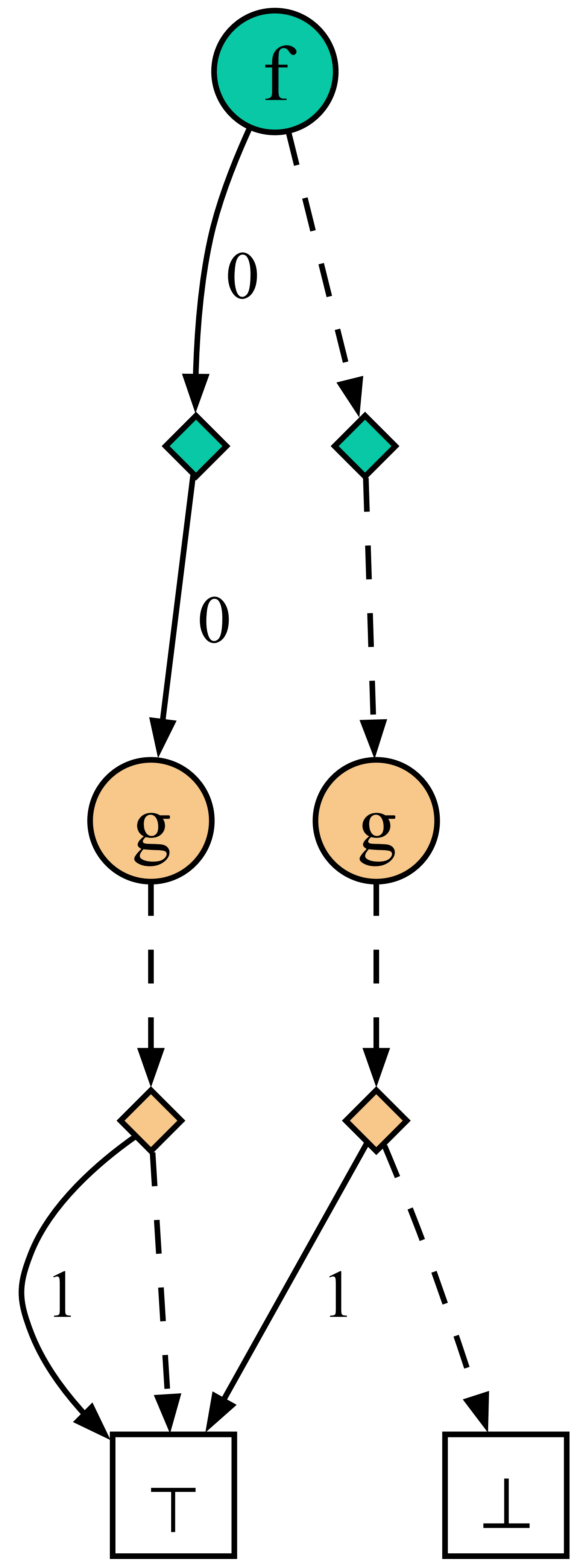
Circle-layers are filters

Diamond-layers are assignments



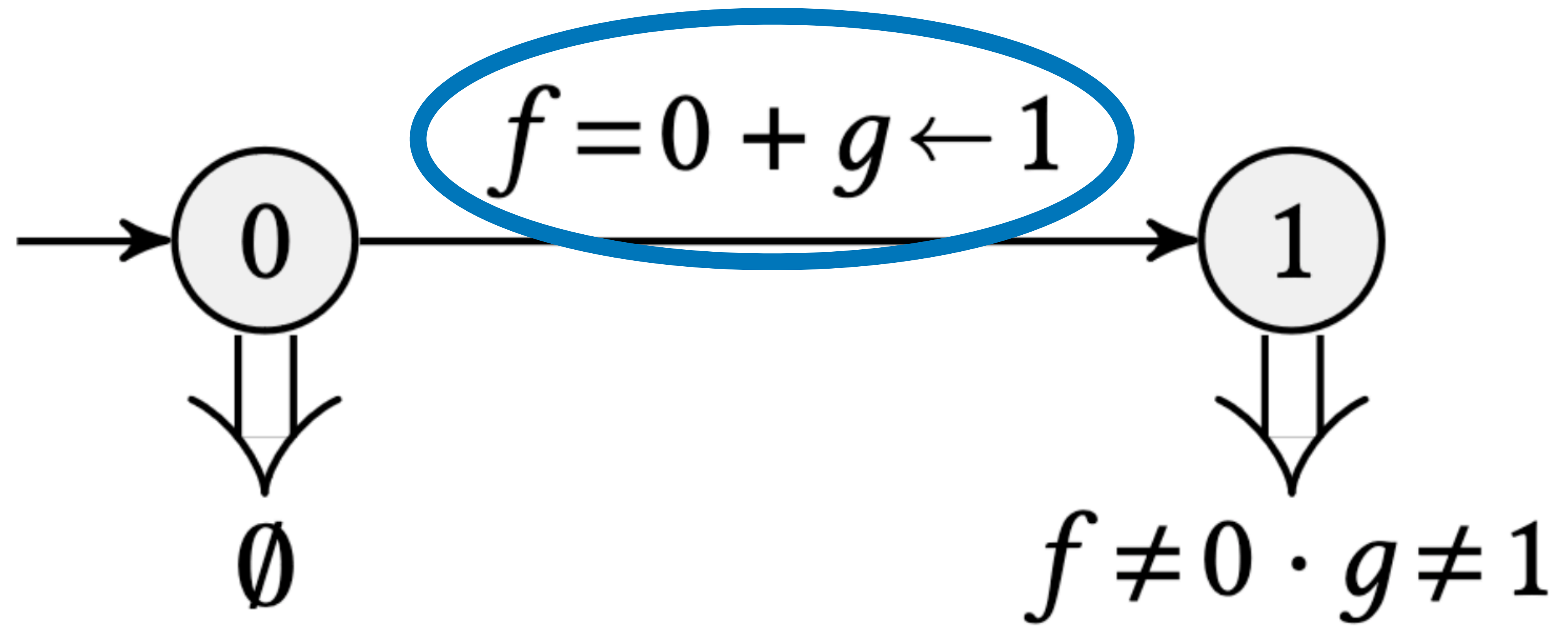
Represent Transitions Symbolically

Symbolic Packet Program (SPP)



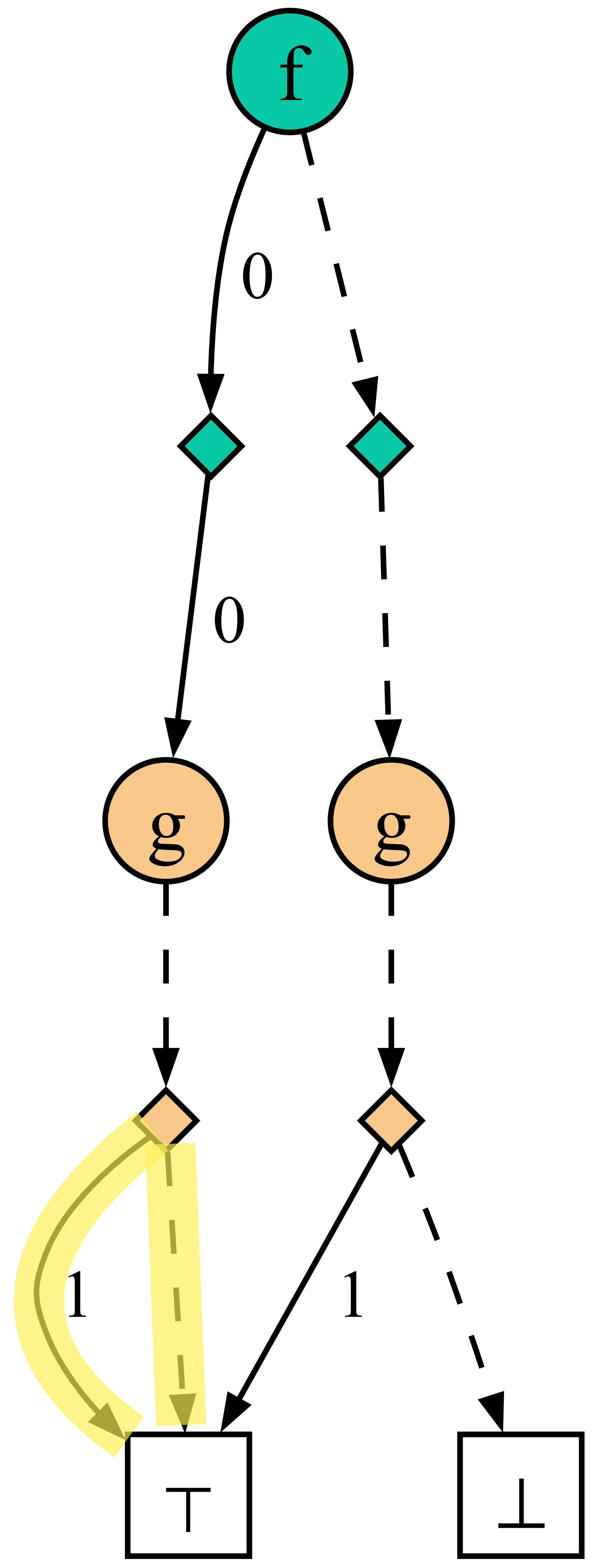
Circle-layers are filters

Diamond-layers are assignments



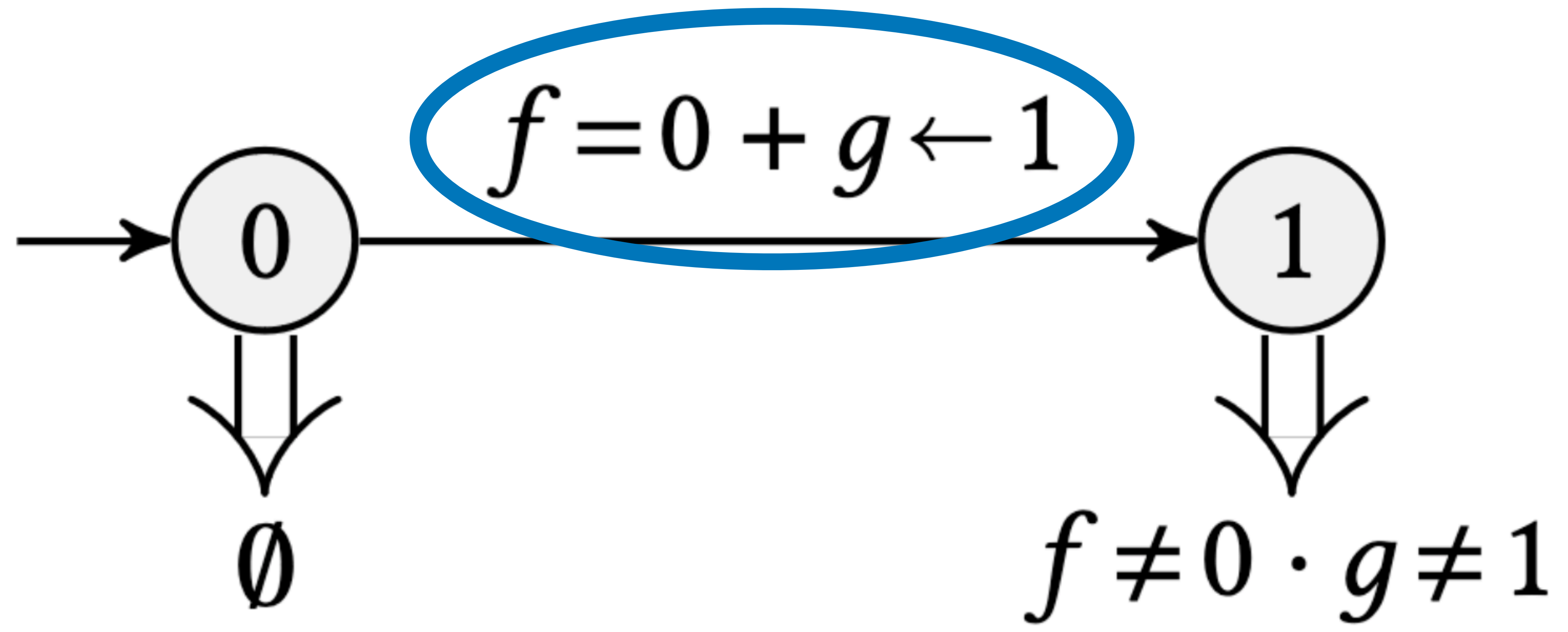
Represent Transitions Symbolically

Symbolic Packet Program (SPP)



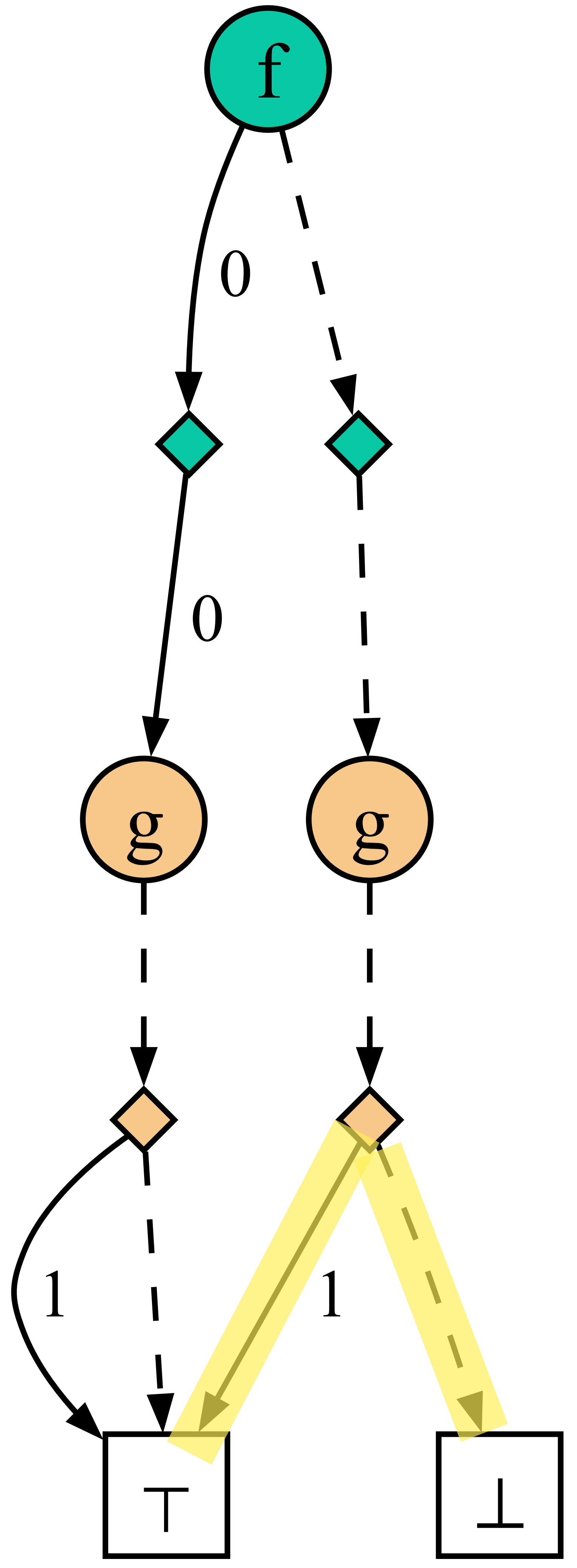
Circle-layers are filters

Diamond-layers are assignments



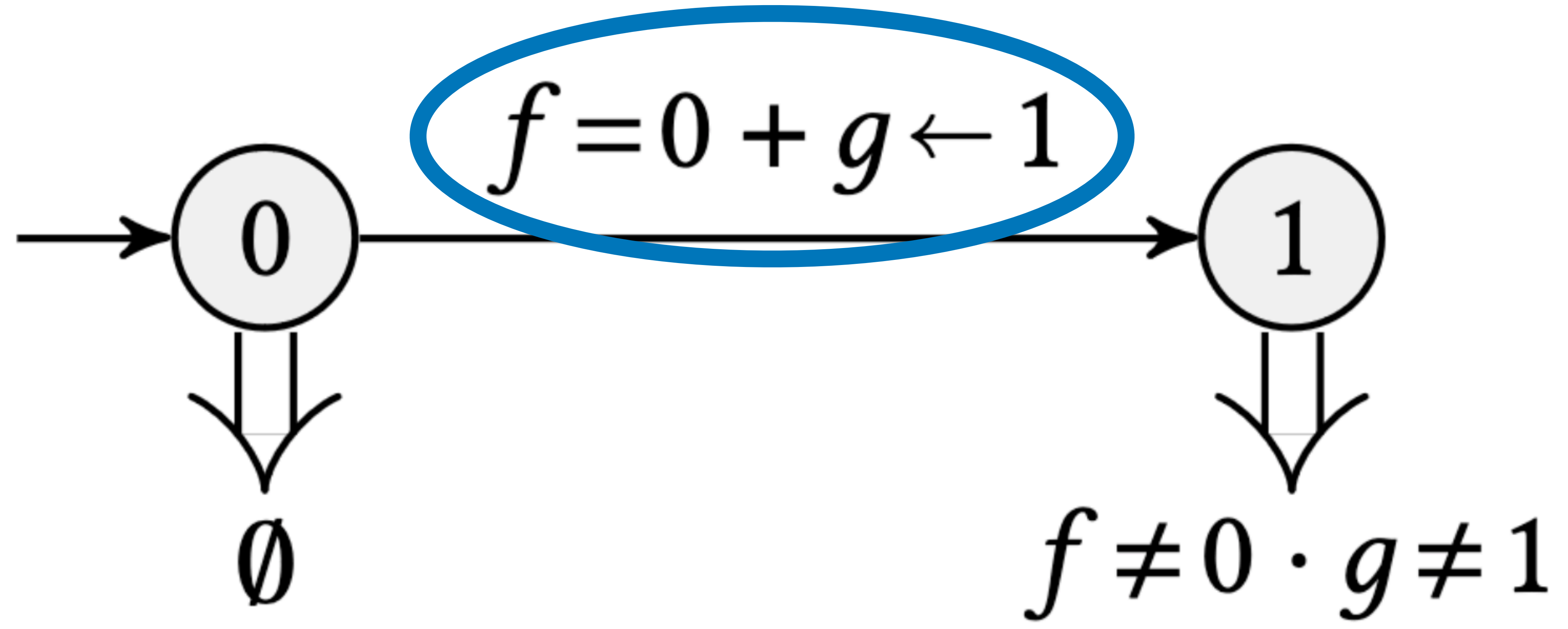
Represent Transitions Symbolically

Symbolic Packet Program (SPP)



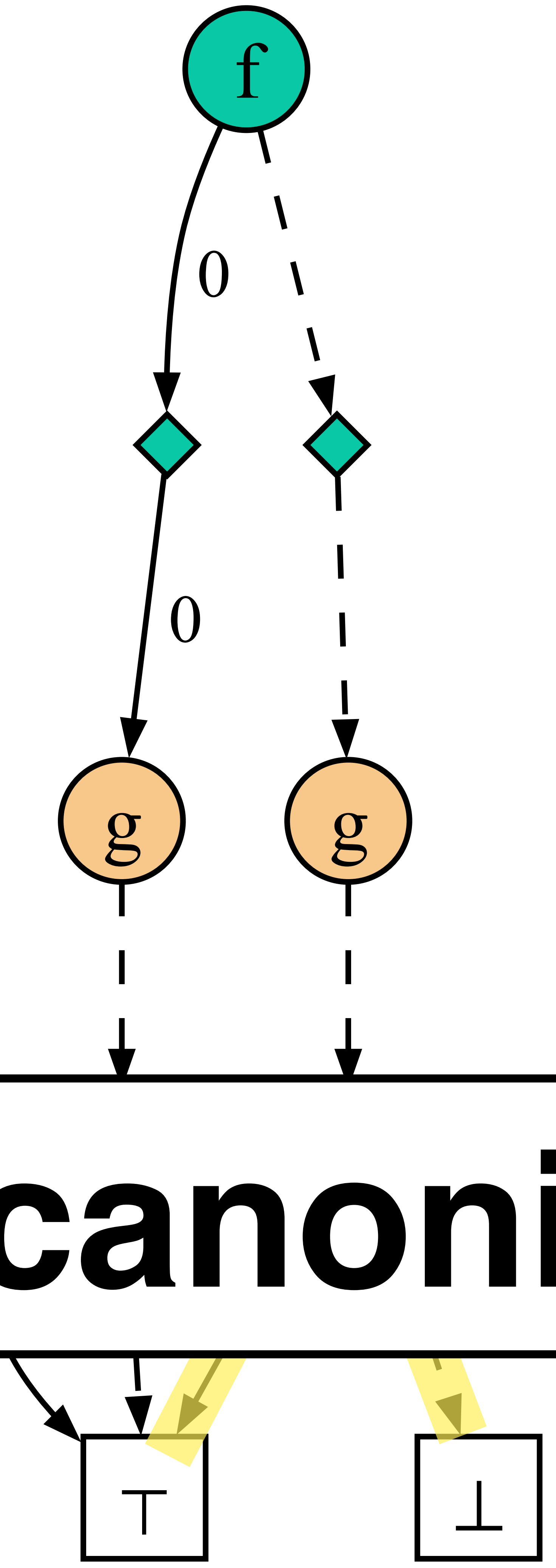
Circle-layers are filters

Diamond-layers are assignments



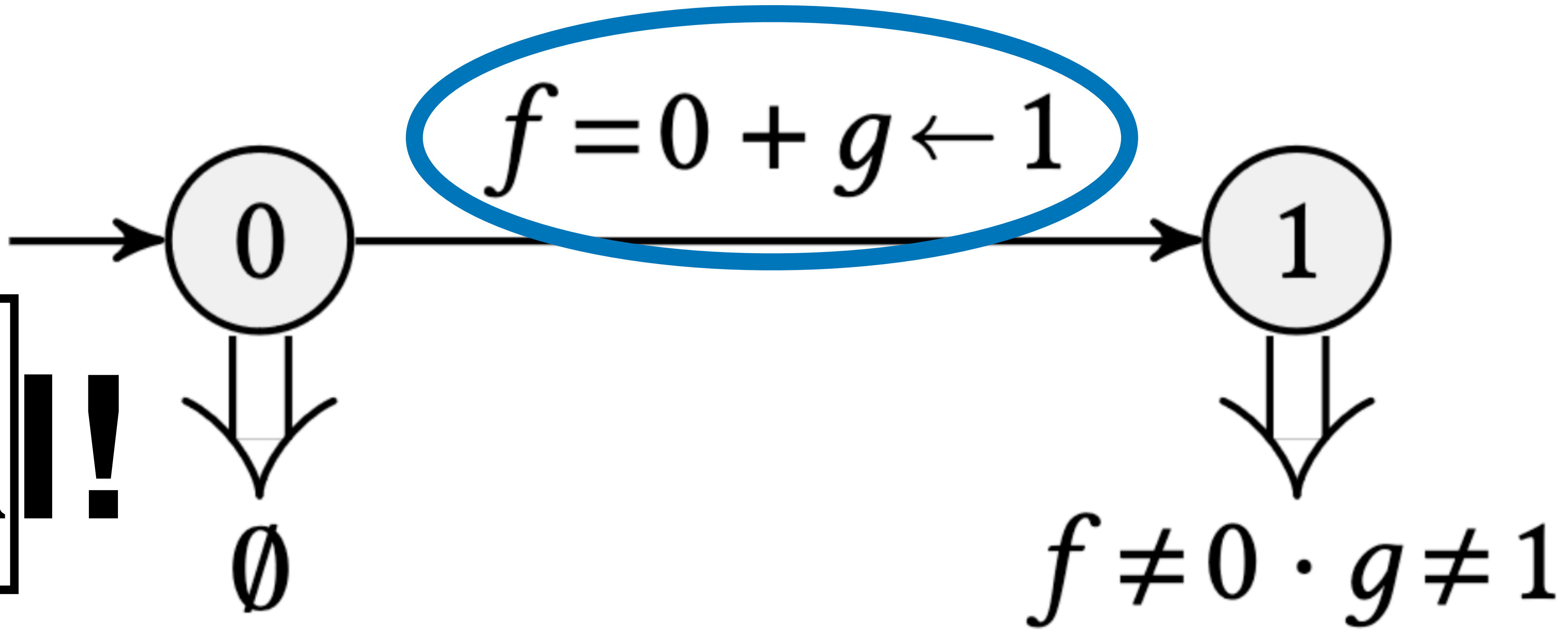
Represent Transitions Symbolically

Symbolic Packet Program (SPP)



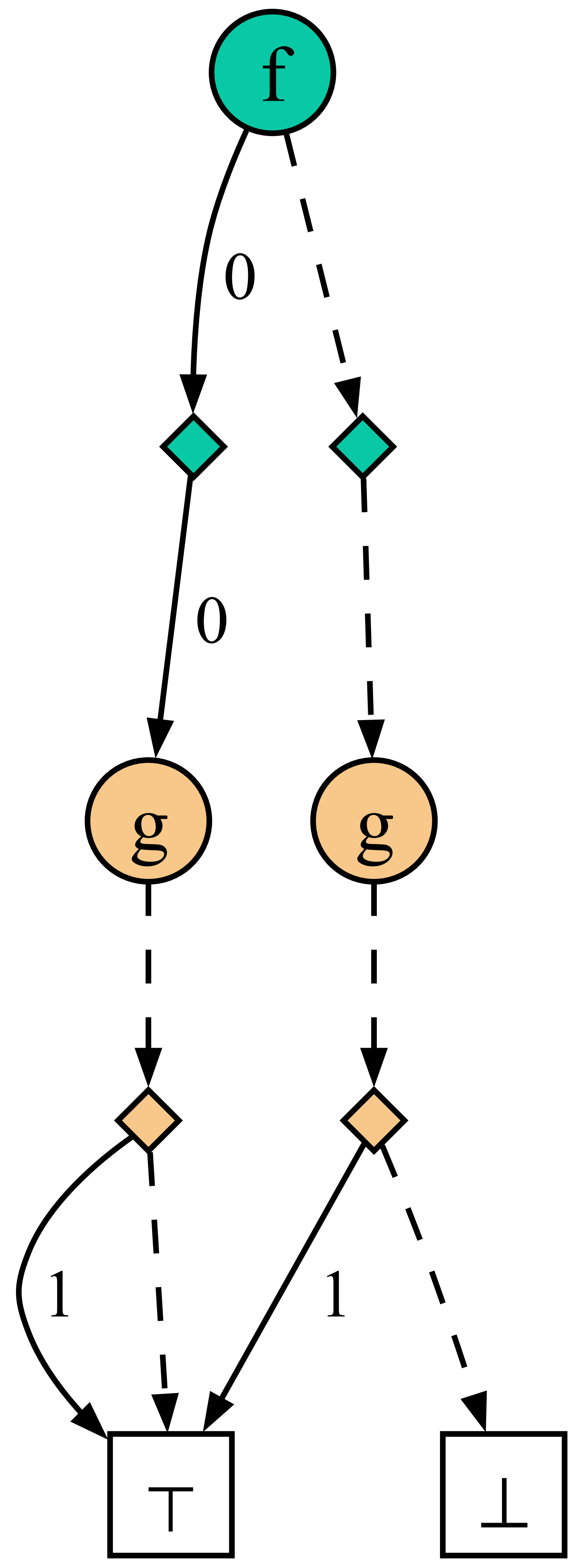
Circle-layers are filters

Diagram-layers are assignments

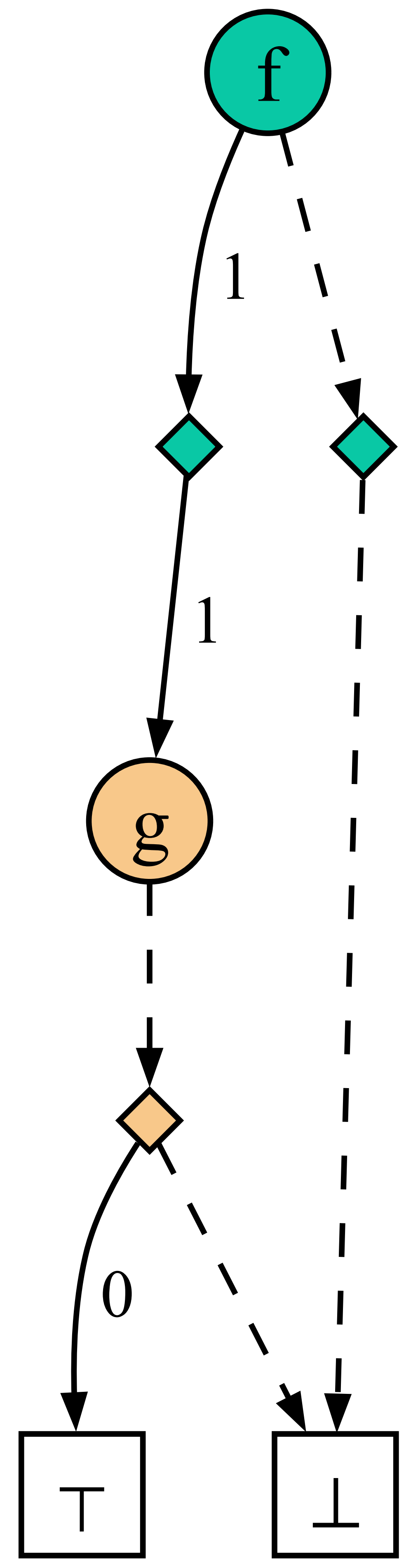


SPPs are canonical!

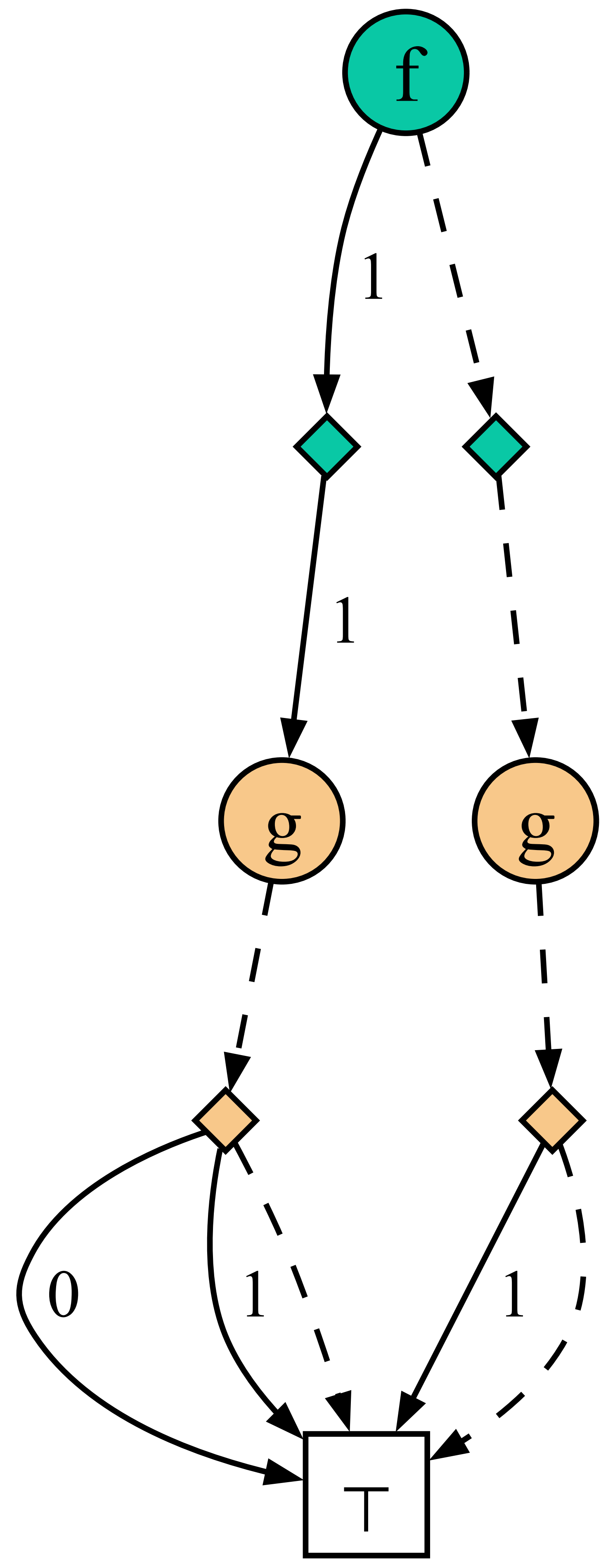
Operations on Symbolic Packet Programs (SPPs)



$p \triangleq f\!=\!0 + g \leftarrow 1$

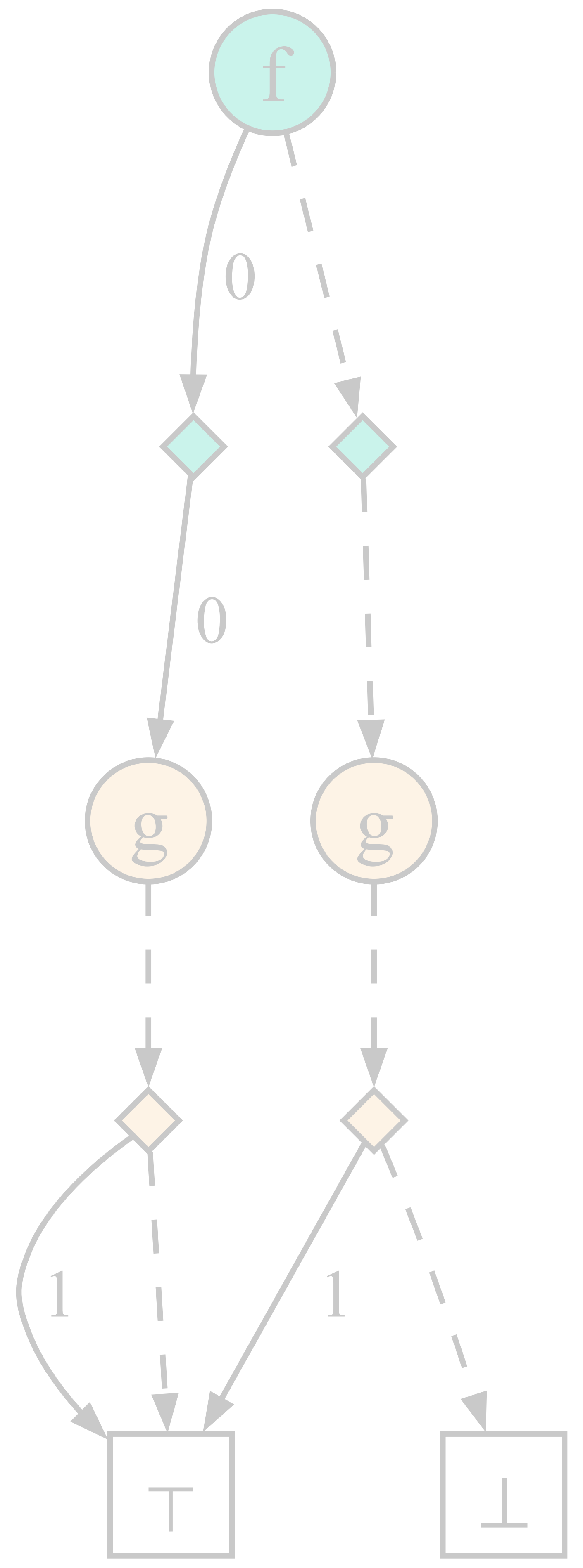


$q \triangleq f\!=\!1 \cdot g \leftarrow 0$

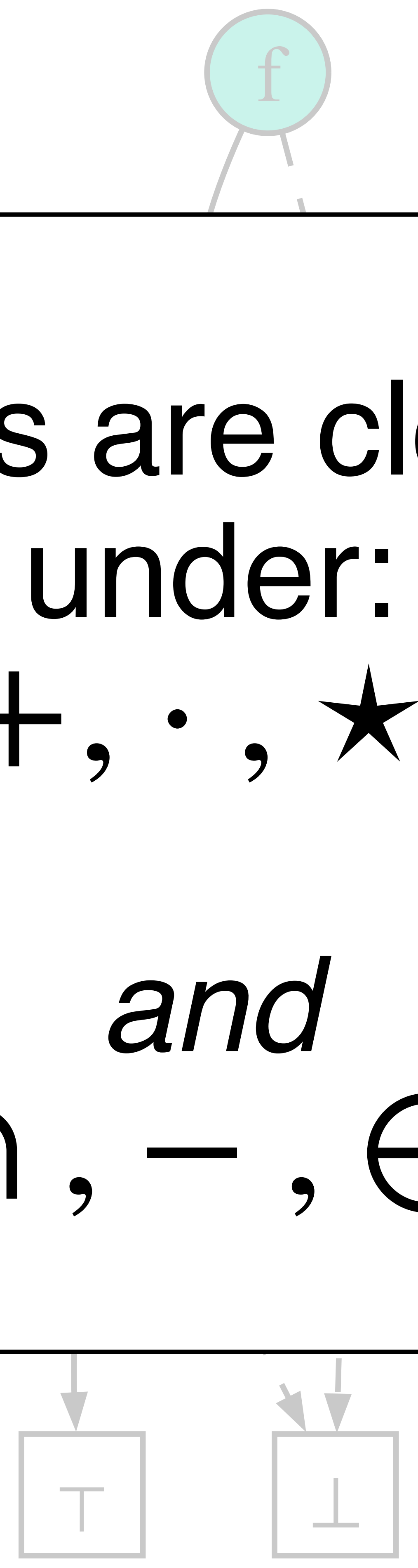


$(p \oplus q)^\star$

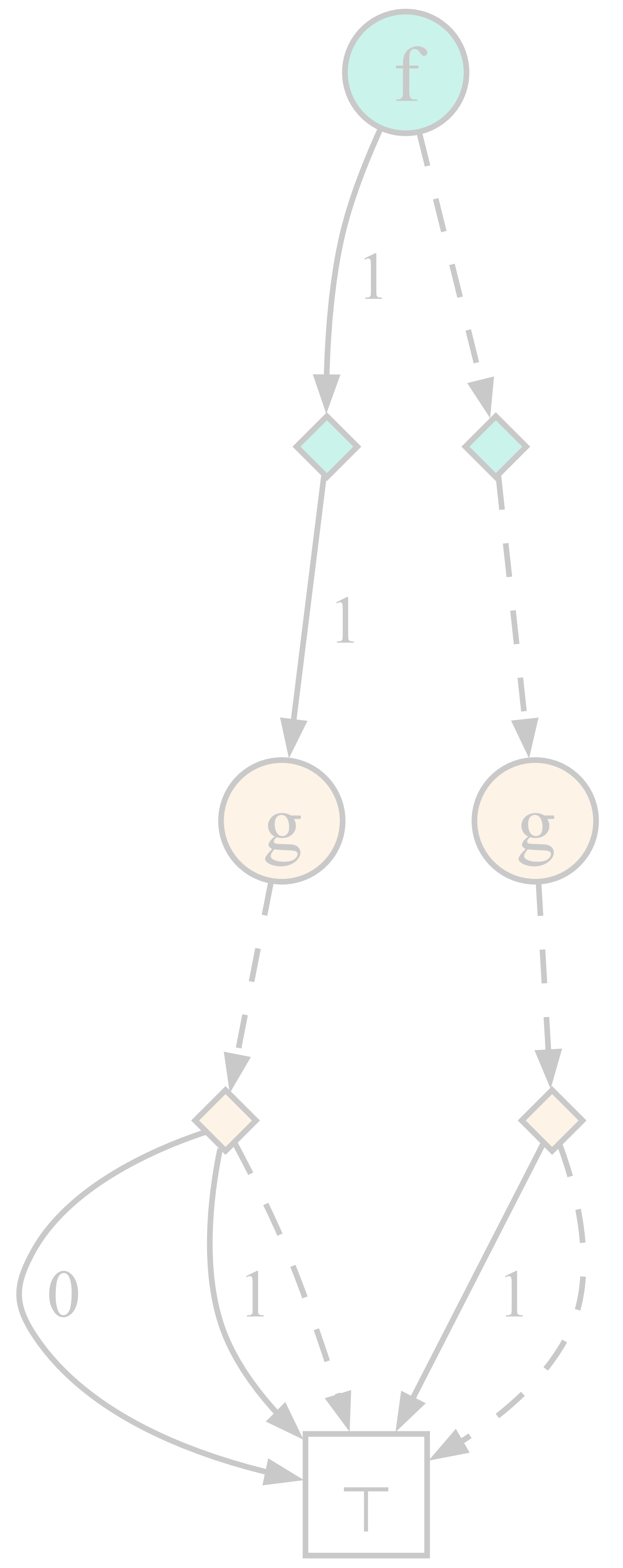
Operations on Symbolic Packet Programs (SPPs)



$p \triangleq f{=}0 + g{\leftarrow}1$



$q \triangleq f{=}1 \cdot g{\leftarrow}0$



$(p \oplus q)^\star$

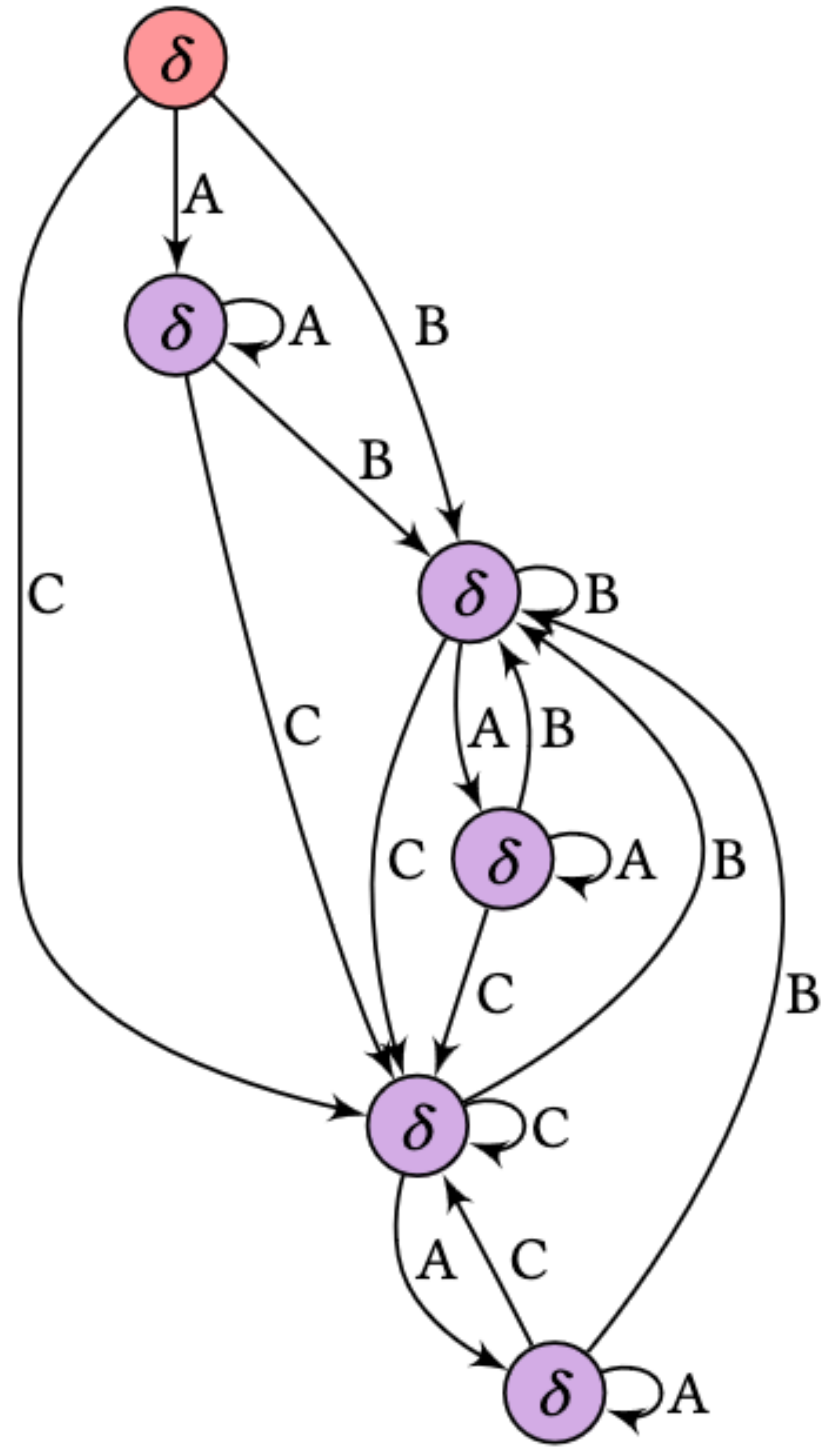
SPPs are closed under:

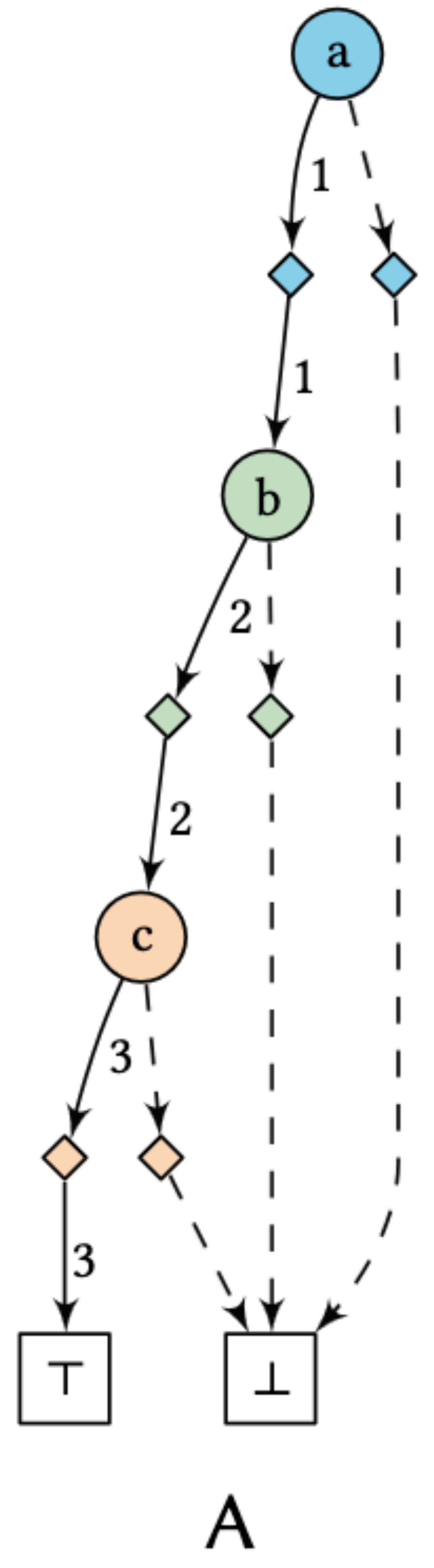
$+, \cdot, \star$

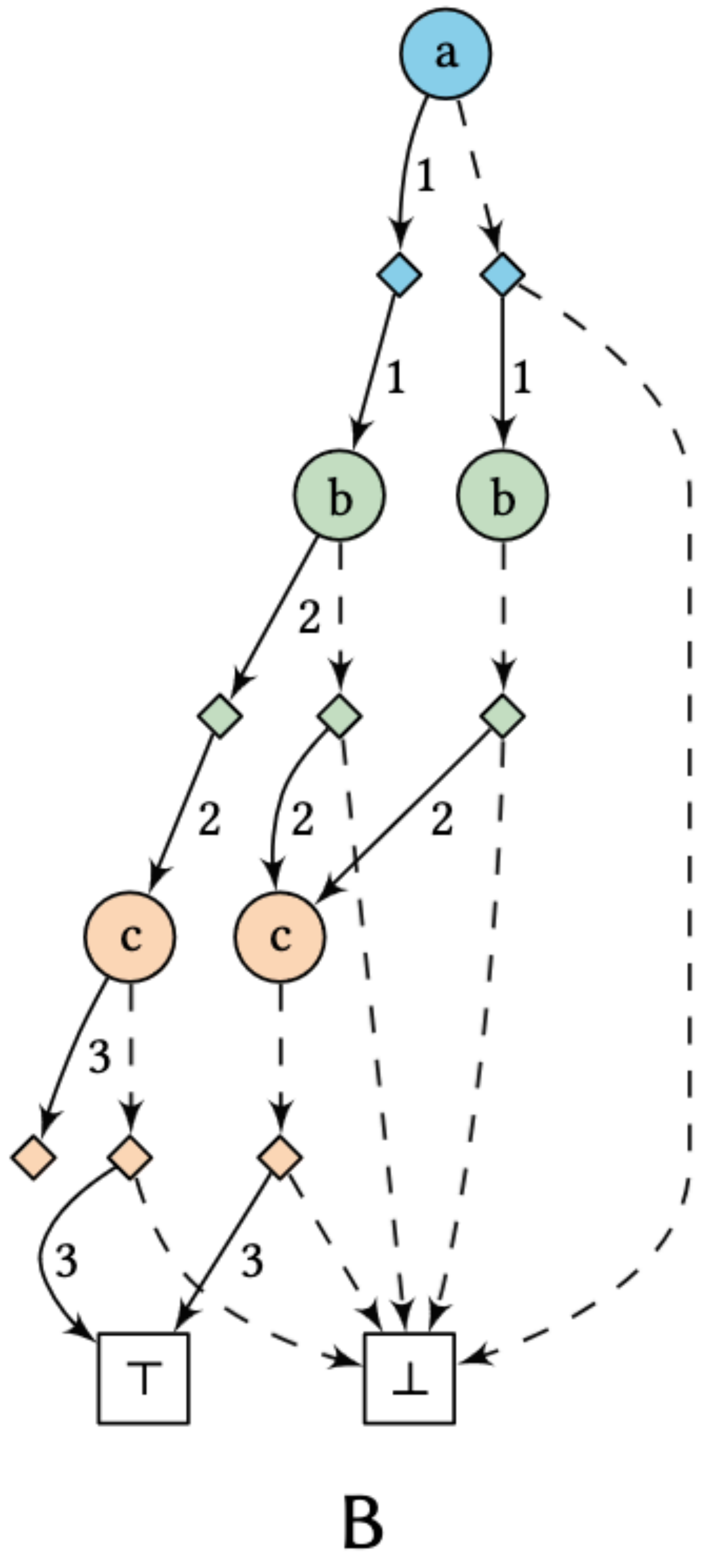
and

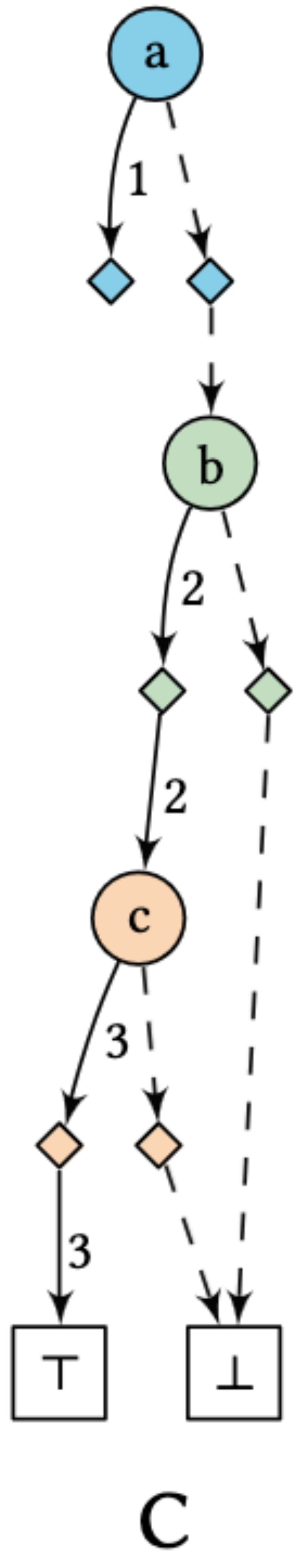
$\cap, -, \oplus$

Symbolic Automaton Construction

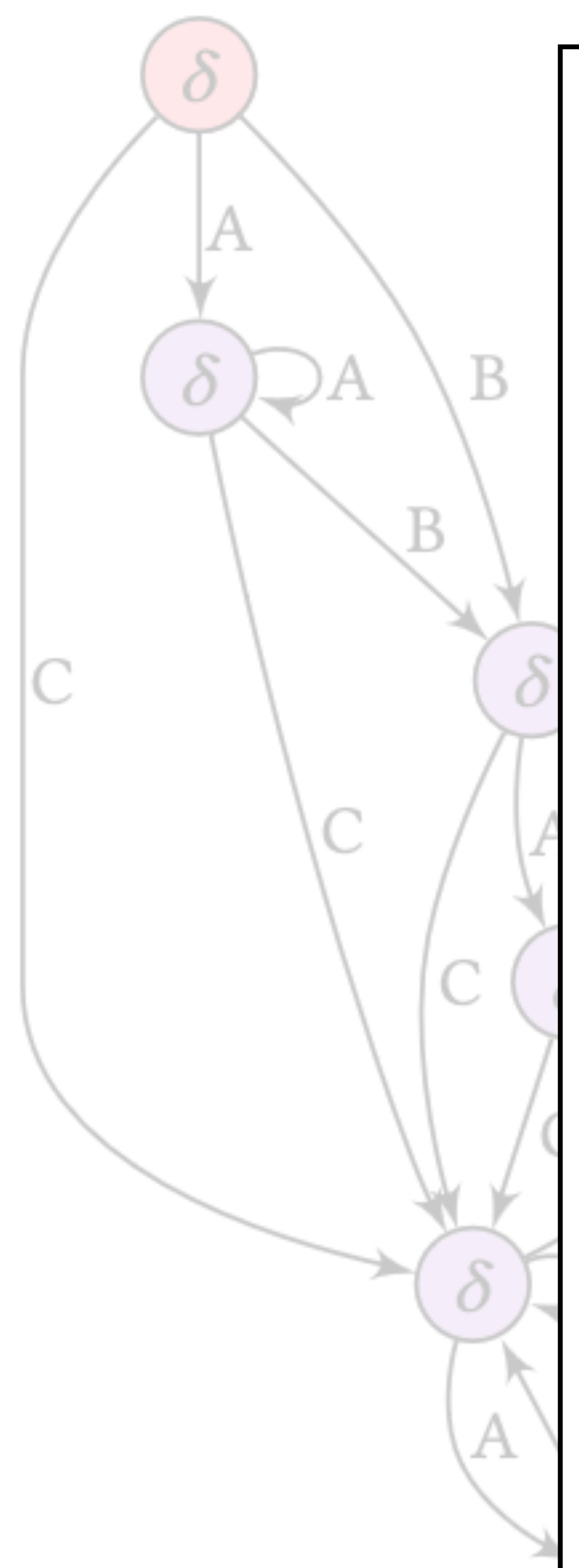


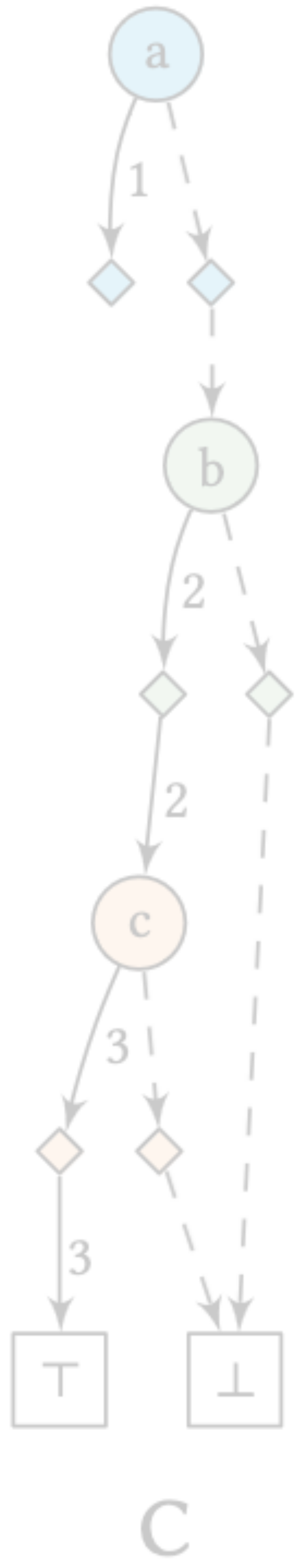






Symbolic Automaton Construction





Automata are also closed under (using SPP operations!):

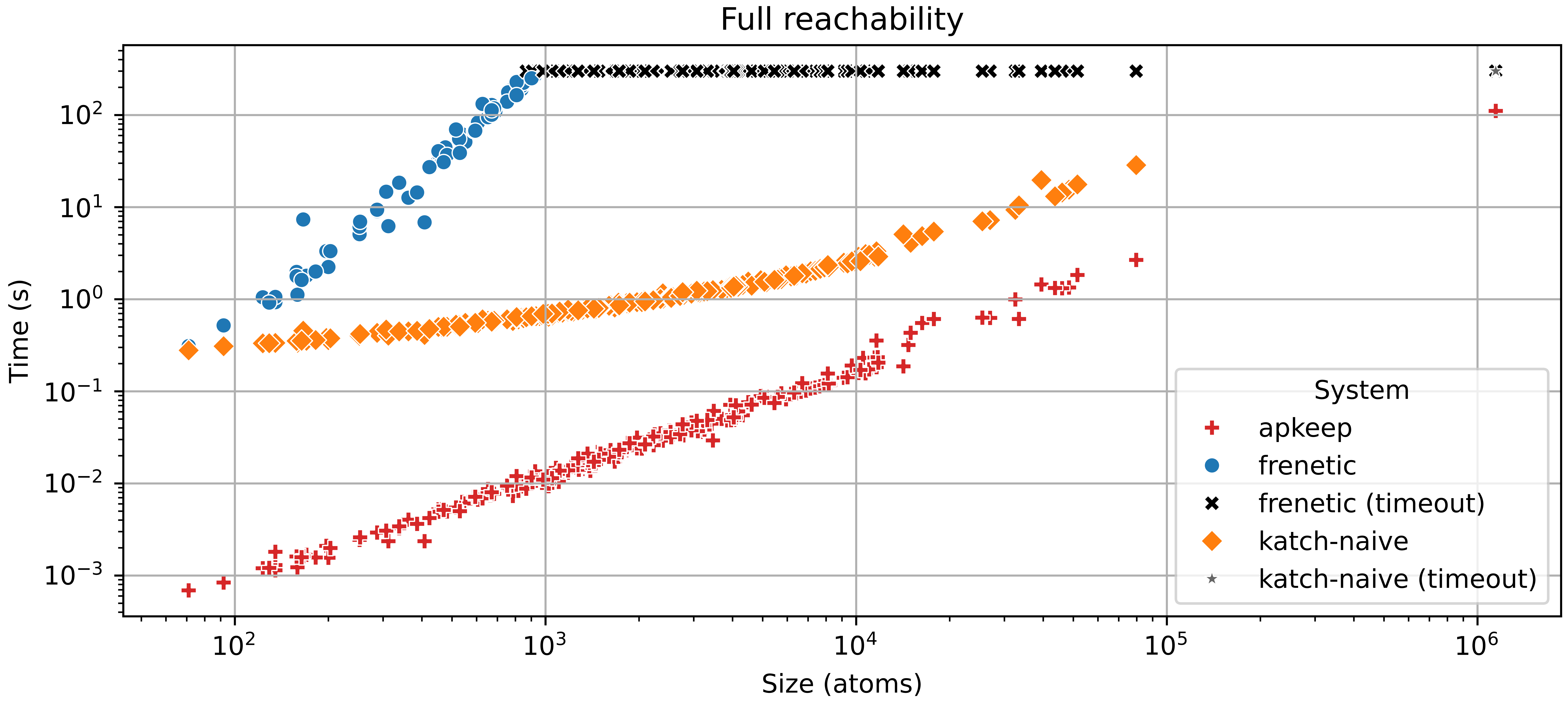
$+, \cdot, \star$

and

$\cap, -, \oplus$

Automata equivalence:

KATch—first attempt



Problem:
NetKAT is limited in practice

Contributions
(This work, PLDI 2024):

1. Symbolic packets and techniques

Problem:
NetKAT is limited in practice

Contributions
(This work, PLDI 2024):

1. Symbolic packets and techniques

2. Extended NetKAT language

NetKAT Programming Language (NKPL)

All-pairs reachability queries, naively:

```
for $i, j \in 1..n$ do
    $\mathsf{sw} = i \cdot \mathsf{net}^\star \cdot \mathsf{sw} = j \not\equiv \emptyset$
```

Number of queries $\propto n^2$

NetKAT Programming Language (NKPL)

All-pairs reachability queries, naively:

```
for $i, j \in 1..n$ do
    $\mathsf{sw} = i \cdot \mathsf{net}^\star \cdot \mathsf{sw} = j \not\equiv \varnothing$
```

Number of queries $\propto n^2$

… and using NKPL features:

```
for $i \in 1..n$ do
    $\mathsf{check}\ (\mathsf{forward}\ (\mathsf{sw}{=}i \cdot \mathsf{net}^\star)) \equiv (\mathsf{sw} \in 1..n)$
```

Number of queries $\propto n$

NetKAT Programming Language (NKPL)

All-pairs reachability queries, naively:

```
for $i, j \in 1..n$ do
    $\mathsf{sw} = i \cdot \mathsf{net}^\star \cdot \mathsf{sw} = j \not\equiv \varnothing$
```

Number of queries $\propto n^2$

… and using NKPL features:

```
for $i \in 1..n$ do
    check $(\mathsf{forward}\ (\mathsf{sw}{=}i \cdot \mathsf{net}^\star)) \equiv (\mathsf{sw} \in 1..n)$
```

Compute output symbolic packet

Number of queries $\propto n$

NetKAT Programming Language (NKPL)

All-pairs reachability queries, naively:

```
for $i, j \in 1..n$ do
    $\mathsf{sw} = i \cdot \mathsf{net}^\star \cdot \mathsf{sw} = j \not\equiv \varnothing$
```

Number of queries $\propto n^2$

… and using NKPL features:

```
for $i \in 1..n$ do
    check $(\text{forward } (\mathsf{sw}=i \cdot \mathsf{net}^\star)) \equiv (\mathsf{sw} \in 1..n)$
```

"Any switch" symbolic packet

Compute output symbolic packet

Number of queries $\propto n$

Evaluation

Full reachability
Time (s)
Size (atoms)
10^{-3}
10^{-2}
10^{-1}
10^{0}
10^{1}
10^{2}
10^{2}
10^{3}
10^{4}
10^{5}
10^{6}
System
apkeep
frenetic
frenetic (timeout)
katch
katch-naive
katch-naive (timeout)

Problem:
NetKAT is limited in practice

Contributions
(This work, PLDI 2024):

1. Symbolic packets and techniques

2. Extended NetKAT language

Problem:
NetKAT is limited in practice

Contributions
(This work, PLDI 2024):

1. Symbolic packets and techniques

2. Extended NetKAT language

3. Symbolic counterexamples

Problem:
NetKAT is limited in practice

Contributions

1. S   es

Automata equivalence:

3. Symbolic counterexamples

KATch: A Fast Symbolic Verifier for NetKAT

1. Symbolic packets and techniques
2. Extended NetKAT language
3. Symbolic counterexamples

Question time

Backup slides

Where is KAT negation?

$$p, q ::= \underline{\bot \mid \top \mid f = v \mid \bigcirc{f \neq v}} \mid f \leftarrow v \mid \text{dup} \mid \underline{p + q \mid p \cdot q} \mid p^\star$$

1. KAT requires arbitrary tests to have negation!

2. We have *only atomic negation,* by preprocessing negations inward using DeMorgan's laws

Brzozowski Derivatives directly from KAT!

$$\begin{array}{llll}
\epsilon(p+q) \triangleq \epsilon(p) \mathbin{\hat{+}} \epsilon(q) & \epsilon(\mathrm{dup}) \triangleq \bot & \delta(p+q) \triangleq \delta(p) \mathbin{\tilde{+}} \delta(q) & \delta(\mathrm{dup}) \triangleq \mathrm{dup} \\
\epsilon(p \cap q) \triangleq \epsilon(p) \mathbin{\hat{\cap}} \epsilon(q) & \epsilon(f=v) \triangleq f=v & \delta(p \cap q) \triangleq \delta(p) \mathbin{\tilde{\cap}} \delta(q) & \delta(f=v) \triangleq \bot \\
\epsilon(p \oplus q) \triangleq \epsilon(p) \mathbin{\hat{\oplus}} \epsilon(q) & \epsilon(f \neq v) \triangleq f \neq v & \delta(p \oplus q) \triangleq \delta(p) \mathbin{\tilde{\oplus}} \delta(q) & \delta(f \neq v) \triangleq \bot \\
\epsilon(p - q) \triangleq \epsilon(p) \mathbin{\hat{-}} \epsilon(q) & \epsilon(f \leftarrow v) \triangleq f \leftarrow v & \delta(p - q) \triangleq \delta(p) \mathbin{\tilde{-}} \delta(q) & \delta(f \leftarrow v) \triangleq \bot \\
\epsilon(p \cdot q) \triangleq \epsilon(p) \mathbin{\hat{\cdot}} \epsilon(q) & \epsilon(\top) \triangleq \top & \delta(p \cdot q) \triangleq \delta(p) \mathbin{\tilde{\cdot}} q \mathbin{\tilde{+}} \epsilon(p) \mathbin{\tilde{\cdot}} \delta(q) & \delta(\top) \triangleq \bot \\
\epsilon(p^{\star}) \triangleq \epsilon(p)^{\star} & \epsilon(\bot) \triangleq \bot & \delta(p^{\star}) \triangleq \epsilon(p)^{\star} \mathbin{\tilde{\cdot}} \delta(p) \mathbin{\tilde{\cdot}} p^{\star} & \delta(\bot) \triangleq \bot
\end{array}$$

Network Verification with NetKAT (POPL 2014)

Network Verification with NetKAT (POPL 2014)

$$p, q ::= \bot \mid \top \mid f = v \mid f \neq v \mid f \leftarrow v \mid \mathrm{dup} \mid p + q \mid p \cdot q \mid p^\star$$

Network Verification with NetKAT (POPL 2014)

$$p, q ::= \bot \mid \top \mid f = v \mid f \neq v \mid f \leftarrow v \mid \mathrm{dup} \mid p + q \mid p \cdot q \mid p^{\star}$$

1. Construct automata for policy and specification.

Network Verification with NetKAT (POPL 2014)

$$p, q ::= \bot \mid \top \mid f = v \mid f \neq v \mid f \leftarrow v \mid \mathrm{dup} \mid p + q \mid p \cdot q \mid p^\star$$

1. Construct automata for policy and specification.

2. Verification is just automata equivalence!

Reachability Example (prior NetKAT)

We can check all-pairs reachability in NetKAT as follows:

$$\mathrm{sw} = 1$$

Reachability Example (prior NetKAT)

We can check all-pairs reachability in NetKAT as follows:

$$\mathsf{sw} = 1 \cdot \mathsf{net}^\star$$

Reachability Example (prior NetKAT)

We can check all-pairs reachability in NetKAT as follows:

$$\mathsf{sw} = 1 \cdot \mathsf{net}^{\star} \cdot \mathsf{sw} = 2$$

Reachability Example (prior NetKAT)

We can check all-pairs reachability in NetKAT as follows:

$$\mathsf{sw} = 1 \cdot \mathsf{net}^{\star} \cdot \mathsf{sw} = 2 \not\equiv \varnothing$$

Reachability Example (prior NetKAT)

We can check all-pairs reachability in NetKAT as follows:

$$\mathsf{sw} = 1 \cdot \mathsf{net}^\star \cdot \mathsf{sw} = 2 \not\equiv \varnothing$$

$$\mathsf{sw} = 1 \cdot \mathsf{net}^\star \cdot \mathsf{sw} = 3 \not\equiv \varnothing$$

$$\mathsf{sw} = 1 \cdot \mathsf{net}^\star \cdot \mathsf{sw} = 4 \not\equiv \varnothing$$

$$\mathsf{sw} = 1 \cdot \mathsf{net}^\star \cdot \mathsf{sw} = 5 \not\equiv \varnothing$$

$$\cdots$$

Reachability Example (prior NetKAT)

We can check all-pairs reachability in NetKAT as follows:

$$\mathsf{sw} = 1 \cdot \mathsf{net}^\star \cdot \mathsf{sw} = 2 \not\equiv \emptyset$$
$$\mathsf{sw} = 1 \cdot \mathsf{net}^\star \cdot \mathsf{sw} = 3 \not\equiv \emptyset$$
$$\mathsf{sw} = 1 \cdot \mathsf{net}^\star \cdot \mathsf{sw} = 4 \not\equiv \emptyset$$
$$\mathsf{sw} = 1 \cdot \mathsf{net}^\star \cdot \mathsf{sw} = 5 \not\equiv \emptyset$$
$$\ldots$$

Requires $O(n^2)$ equivalence queries

NetKAT Programming Language (NKPL)

All-pairs reachability queries, naively:

$$\boxed{\begin{array}{c} \mathsf{sw} = 1 \cdot \mathsf{net}^\star \cdot \mathsf{sw} = 2 \not\equiv \emptyset \\ \mathsf{sw} = 1 \cdot \mathsf{net}^\star \cdot \mathsf{sw} = 3 \not\equiv \emptyset \\ \mathsf{sw} = 1 \cdot \mathsf{net}^\star \cdot \mathsf{sw} = 4 \not\equiv \emptyset \\ \bullet\bullet\bullet \end{array}}$$

NetKAT Programming Language (NKPL)

All-pairs reachability queries, naively:

$$\mathsf{sw} = 1 \cdot \mathsf{net}^\star \cdot \mathsf{sw} = 2 \not\equiv \emptyset$$
$$\mathsf{sw} = 1 \cdot \mathsf{net}^\star \cdot \mathsf{sw} = 3 \not\equiv \emptyset$$
$$\mathsf{sw} = 1 \cdot \mathsf{net}^\star \cdot \mathsf{sw} = 4 \not\equiv \emptyset$$

• • •

… and using NKPL features:

```
for $i \in 1..n$ do
    check (forward ($\mathsf{sw}{=}i \cdot \mathsf{net}^\star$)) $\equiv$ ($\mathsf{sw} \in 1..n$)
```

NetKAT Programming Language (NKPL)

All-pairs reachability queries, naively:

$$\mathsf{sw} = 1 \cdot \mathsf{net}^\star \cdot \mathsf{sw} = 2 \not\equiv \varnothing$$
$$\mathsf{sw} = 1 \cdot \mathsf{net}^\star \cdot \mathsf{sw} = 3 \not\equiv \varnothing$$
$$\mathsf{sw} = 1 \cdot \mathsf{net}^\star \cdot \mathsf{sw} = 4 \not\equiv \varnothing$$
$$\cdots$$

… and using NKPL features:

```
for $i \in 1..n$ do
    check (forward ($\mathsf{sw}=i \cdot \mathsf{net}^\star$)) $\equiv$ ($\mathsf{sw} \in 1..n$)
```

Compute output symbolic packet

NetKAT Programming Language (NKPL)

All-pairs reachability queries, naively:

$$\mathsf{sw} = 1 \cdot \mathsf{net}^\star \cdot \mathsf{sw} = 2 \not\equiv \emptyset$$
$$\mathsf{sw} = 1 \cdot \mathsf{net}^\star \cdot \mathsf{sw} = 3 \not\equiv \emptyset$$
$$\mathsf{sw} = 1 \cdot \mathsf{net}^\star \cdot \mathsf{sw} = 4 \not\equiv \emptyset$$
$$\cdots$$

… and using NKPL features:

```
for $i \in 1..n$ do
    check $(\mathsf{forward}\ (\mathsf{sw}{=}i \cdot \mathsf{net}^\star)) \equiv (\mathsf{sw} \in 1..n)$
```

"Any switch" symbolic packet

Compute output symbolic packet

NetKAT Programming Language (NKPL)

All-pairs reachability queries, naively:

$$\mathsf{sw} = 1 \cdot \mathsf{net}^\star \cdot \mathsf{sw} = 2 \not\equiv \emptyset$$
$$\mathsf{sw} = 1 \cdot \mathsf{net}^\star \cdot \mathsf{sw} = 3 \not\equiv \emptyset$$
$$\mathsf{sw} = 1 \cdot \mathsf{net}^\star \cdot \mathsf{sw} = 4 \not\equiv \emptyset$$
$$\bullet\bullet\bullet$$

… and using NKPL features:

```
for $i \in 1..n$ do
    check $(\text{forward}\ (\mathsf{sw}{=}i \cdot \mathsf{net}^\star)) \equiv (\mathsf{sw} \in 1..n)$
```

"Any switch" symbolic packet

Compute output symbolic packet

Each query is equivalent to *n* original queries — requiring only O(n) queries!

NetKAT Programming Language (NKPL)

Expressions

forward e, backward e

$e_1 \cap e_2$, $e_1 \oplus e_2$, $e_1 - e_2$

$\hat{\exists} f\ e$, $\hat{\forall} f\ e$

Statements

check $e_1 \equiv e_2$

check $e_1 \not\equiv e_2$

print e

$x = e$

for $i \in n_1..n_2$ do c

NetKAT Programming Language (NKPL)

Expressions

forward e, backward e

$e_1 \cap e_2$, $e_1 \oplus e_2$, $e_1 - e_2$

$\hat{\exists} f\ e$, $\hat{\forall} f\ e$

Statements

check $e_1 \equiv e_2$
check $e_1 \not\equiv e_2$
print e
$x = e$
for $i \in n_1..n_2$ do c

All-pairs reachability queries, naively:

$$\boxed{\begin{array}{c} \mathsf{sw} = 1 \cdot \mathsf{net}^\star \cdot \mathsf{sw} = 2 \not\equiv \emptyset \\ \mathsf{sw} = 1 \cdot \mathsf{net}^\star \cdot \mathsf{sw} = 3 \not\equiv \emptyset \\ \mathsf{sw} = 1 \cdot \mathsf{net}^\star \cdot \mathsf{sw} = 4 \not\equiv \emptyset \\ \cdots \end{array}}$$

NetKAT Programming Language (NKPL)

Expressions

forward e, backward e

$e_1 \cap e_2,\ \ e_1 \oplus e_2,\ \ e_1 - e_2$

$\hat{\exists} f\ e,\ \hat{\forall} f\ e$

Statements

check $e_1 \equiv e_2$

check $e_1 \not\equiv e_2$

print e

$x = e$

for $i \in n_1..n_2$ do c

All-pairs reachability queries, naively:

$$\mathsf{sw} = 1 \cdot \mathsf{net}^\star \cdot \mathsf{sw} = 2 \not\equiv \emptyset$$
$$\mathsf{sw} = 1 \cdot \mathsf{net}^\star \cdot \mathsf{sw} = 3 \not\equiv \emptyset$$
$$\mathsf{sw} = 1 \cdot \mathsf{net}^\star \cdot \mathsf{sw} = 4 \not\equiv \emptyset$$

• • •

… and using NKPL features:

```
for $i \in 1..n$ do
    check (forward (sw$=i \cdot$ net$^\star$)) $\equiv$ (sw $\in 1..n$)
```