Insights into the accuracy of algorithms for ASPA based route leak detection

K. Sriram and Jakob Heitz
ksriram@nist.gov   jheitz@cisco.com

SIDROPS WG Meeting
IETF 110
March 2021

Acknowledgements: Authors wish to thank Ties de Kock (RIPE), Alexander Azimov (Yandex), Ben Maddison (Workonline Communications), and Kyehwan Lee (NIST) for review and comments.

Overview – Key Takeaways

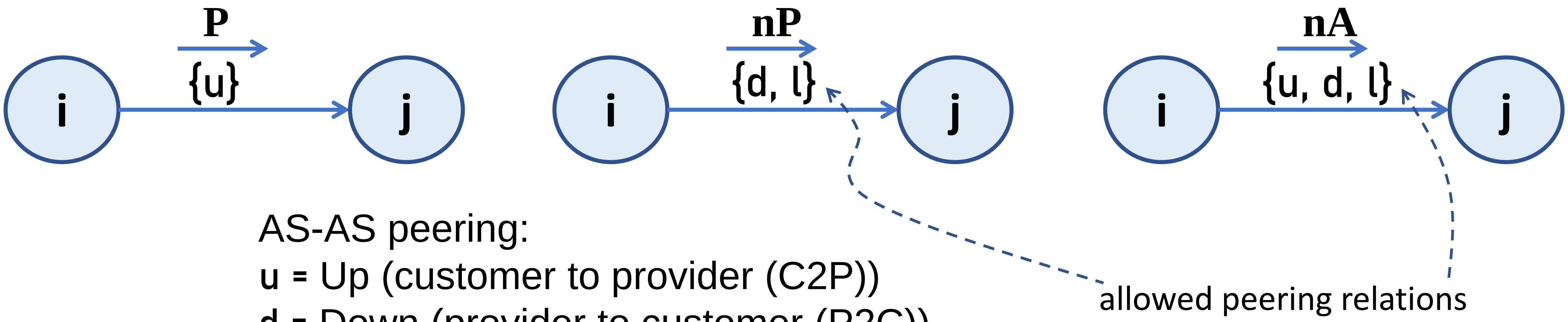
- Downstream AS path algorithm in the current draft (-06) has an oversight
  - ➢ Some Valid AS paths are misclassified as Unknown
- A correct algorithm exists with formal proof
  - ➢ Classifies all Valid, Invalid, and Unknown AS paths correctly
- We think the proposed new algorithm is correct and efficient
  - ➢ Minimizes ASPA look ups
- We recommend updating the algorithm in the draft

ASPA Hop Check Function

Definition:

$$g(AS(i), AS(j)) = \begin{cases} \mathbf{P} & \text{if AS(i) attests AS(j) is a provider} \\ \mathbf{nP} & \text{if AS(i) attests AS(j) is not a provider} \\ \mathbf{nA} & \text{if AS(i) does not have an ASPA} \end{cases}$$

P: Provider
nP: not Provider
nA: no Attestation



AS-AS peering:
u = Up (customer to provider (C2P))
d = Down (provider to customer (P2C))
l = Lateral (peer to peer (p2p))

Note: It is well understood that ASPAs are AFI dependent, so AFI is not explicitly shown in function g for simplicity. In actual implementation, the g function would include the AFI: g(AS(i), AS(j), AFI).

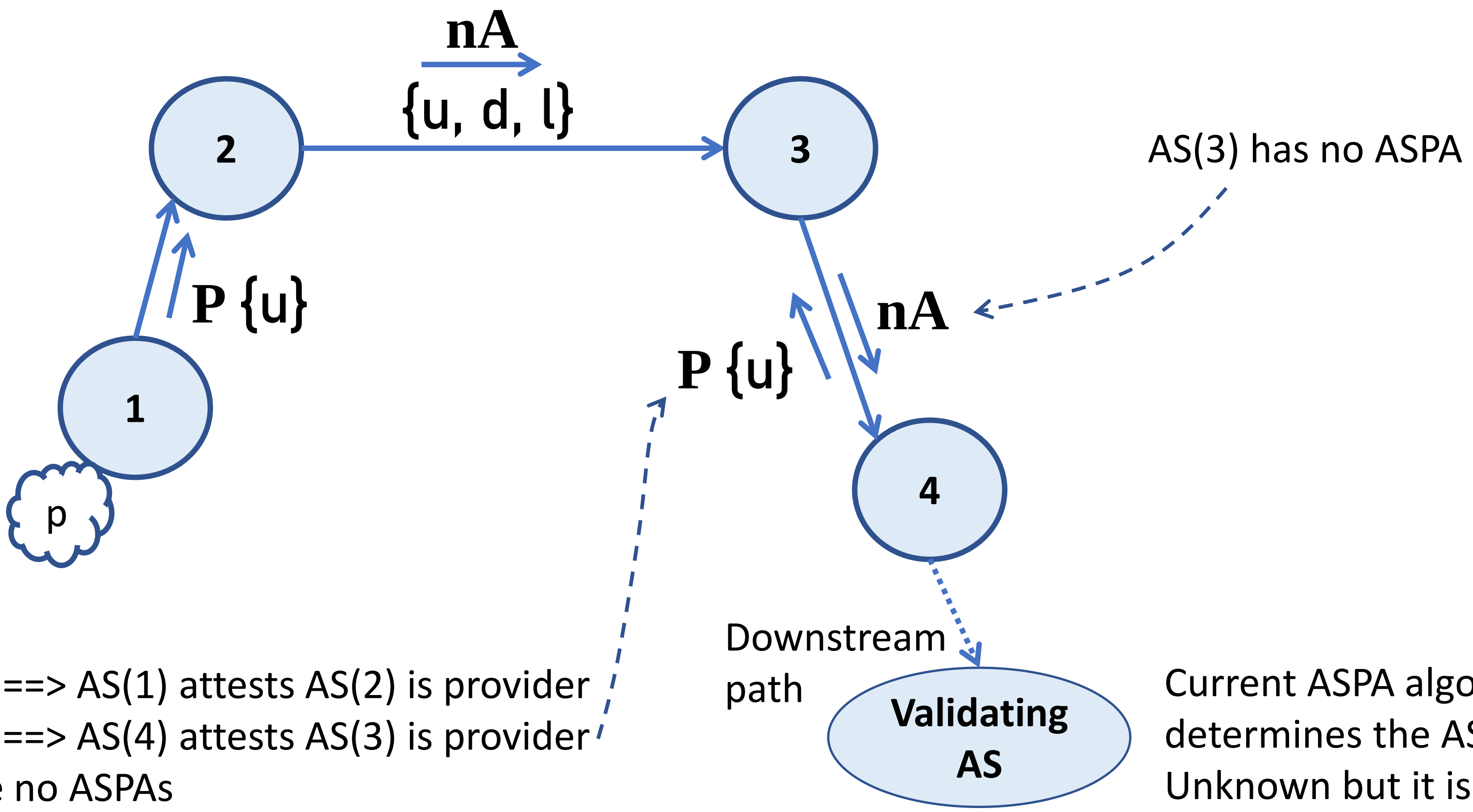
An error in the current draft-06 algorithm

Example 1

- The current algorithm classifies some Valid downstream AS paths as Unknown

ASPA hop check:
P: Provider
nP: not Provider
nA: no Attestation

AS-AS peering:
u = Up
d = Down
l = Lateral

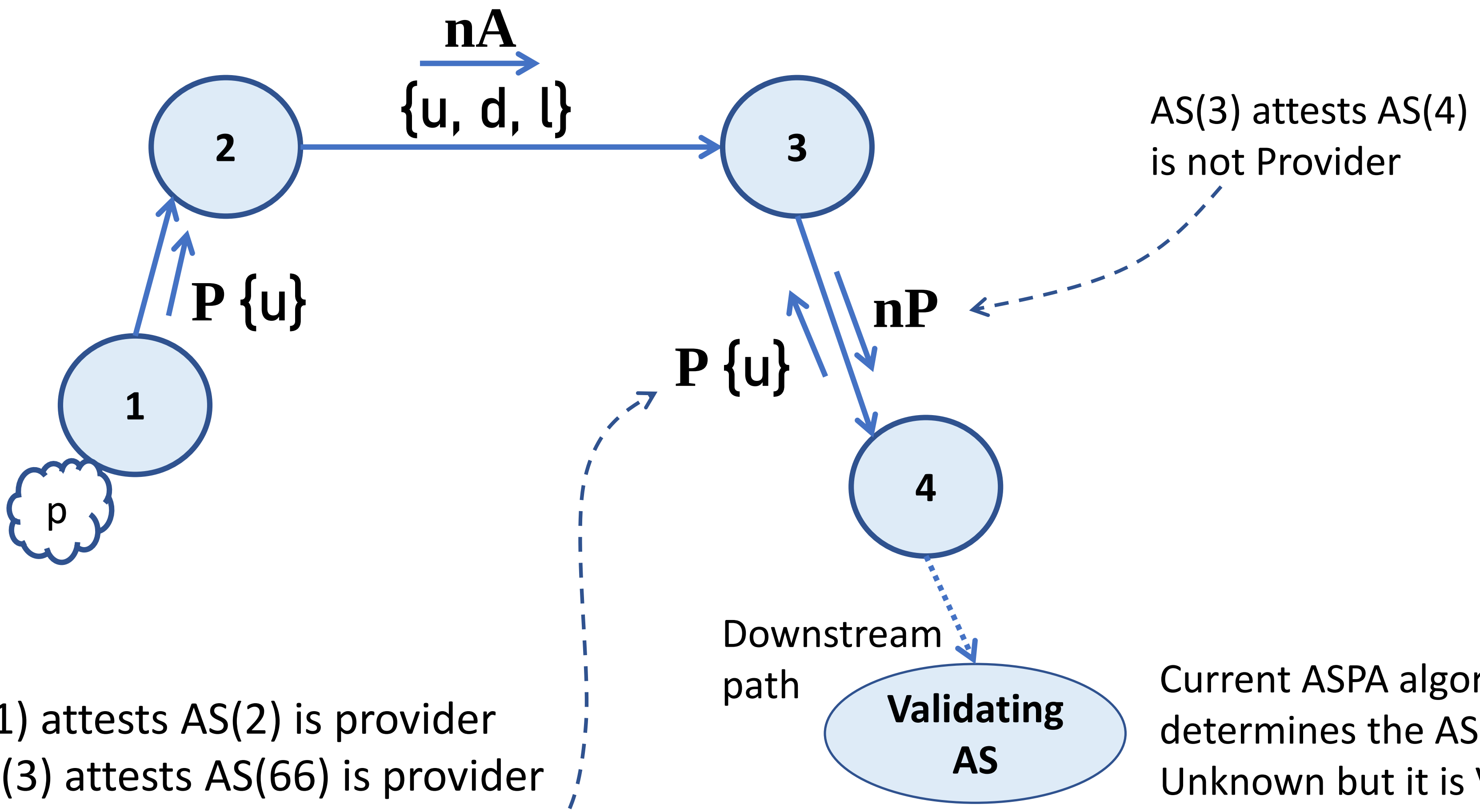


ASPA: {AS(1), AS(2)} ==> AS(1) attests AS(2) is provider
ASPA: {AS(4), AS(3)} ==> AS(4) attests AS(3) is provider
AS(2) and AS(3) have no ASPAs

Current ASPA algorithm determines the AS path to be Unknown but it is Valid

An error in the current draft-06 algorithm

Example 2

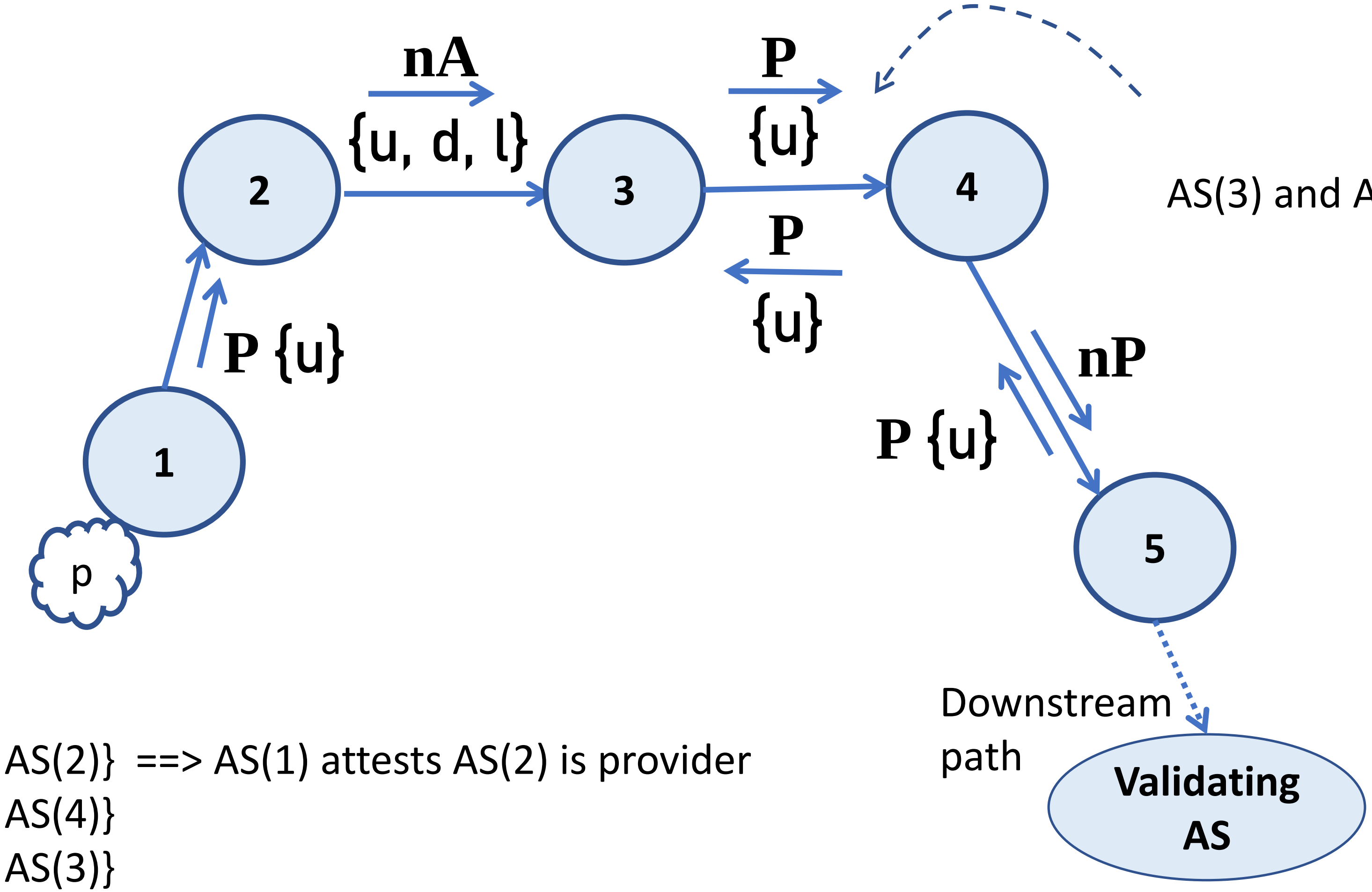


Current ASPA algorithm determines the AS path to be Unknown but it is Valid

ASPAs:
{AS(1), AS(2)} ==> AS(1) attests AS(2) is provider
{AS(3), AS(66)} ==> AS(3) attests AS(66) is provider
{AS(4), AS(3)} ==> AS(4) attests AS(3) is provider
AS(2) and AS(3) have no ASPAs

An error in the current draft-06 algorithm

Example 3



AS(3) and AS(4) are siblings

ASPAs:
{AS(1), AS(2)} ==> AS(1) attests AS(2) is provider
{AS(3), AS(4)}
{AS(4), AS(3)}
{AS(5), AS(4)}
AS(2) doesn't have ASPA

Current ASPA algorithm determines the path to be Unknown but it is Valid

Design principles for ASPA based for route leak detection:

Focusing on Downstream Path only

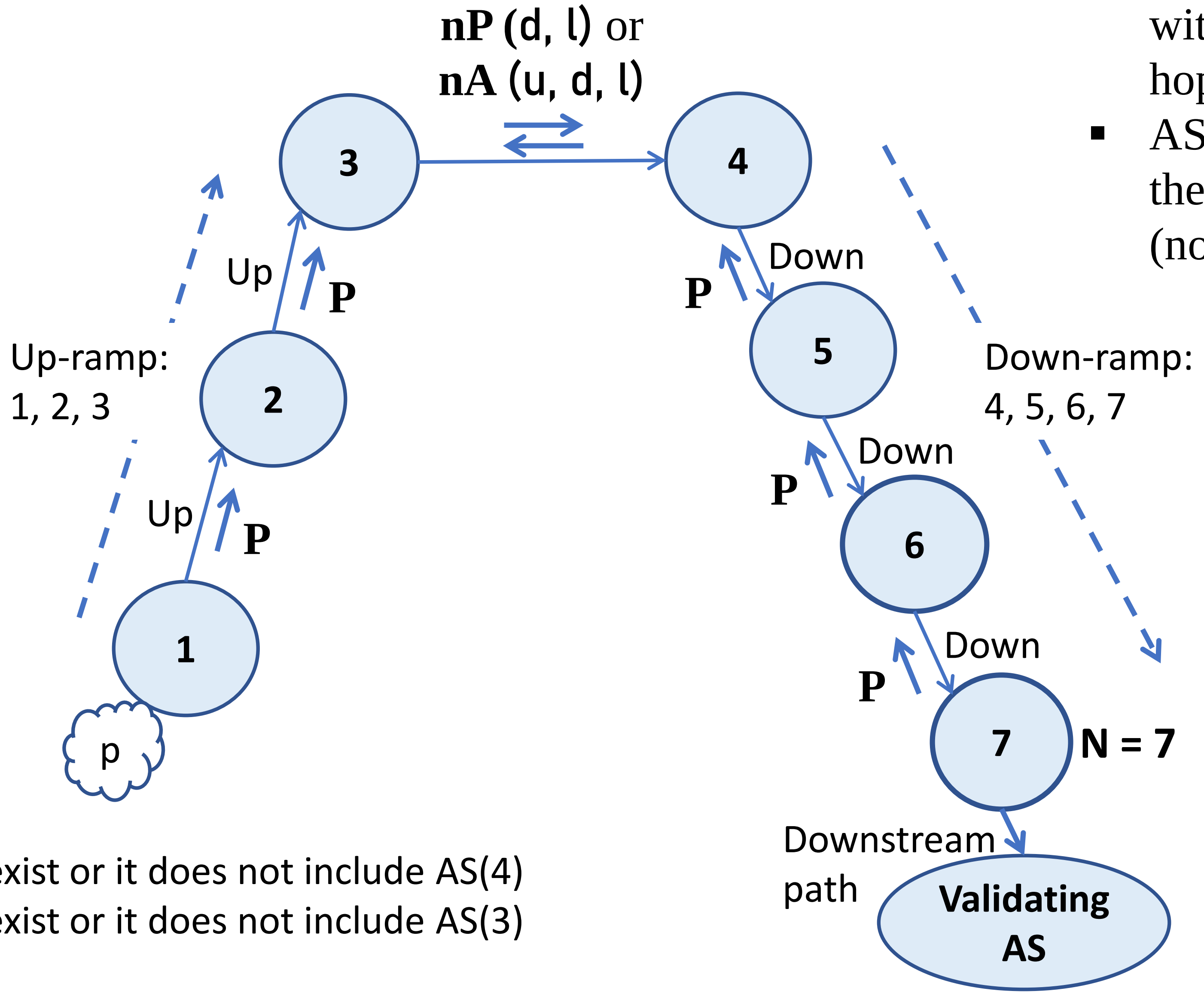
- AS path is Valid if
  - There is a up-ramp of Valid C2P hops on the left,
  - There is a down-ramp of Valid P2C hops on the right, and
  - Either no hops in the middle or a single hop which is **nP** or **nA**.
  - Up-ramp or down-ramp or both can be absent
    - When both are absent, the AS path is a single hop that is **nP** or **nA**.
- In effect, the above can be also stated as follows: If every transit AS has at least one neighbor that attests it a provider, then the AS path is valid.
- If the AS path segment in the middle (between the up-ramp and the down-ramp) is 2 or more hops long, then the AS path can be only Invalid or Unknown:
  - If there are opposing valley walls, i.e., an **nP** hop from left to right and a subsequent **nP** hop from right to left, then no matter what is in between, there must be at least one valley in the AS path and hence it is Invalid.
  - Otherwise, the AS path is Unknown.

Valid downstream AS path

ASPA hop check:
P: Provider
nP: not Provider
nA: no Attestation

AS-AS peering:
u = Up
d = Down
l = Lateral

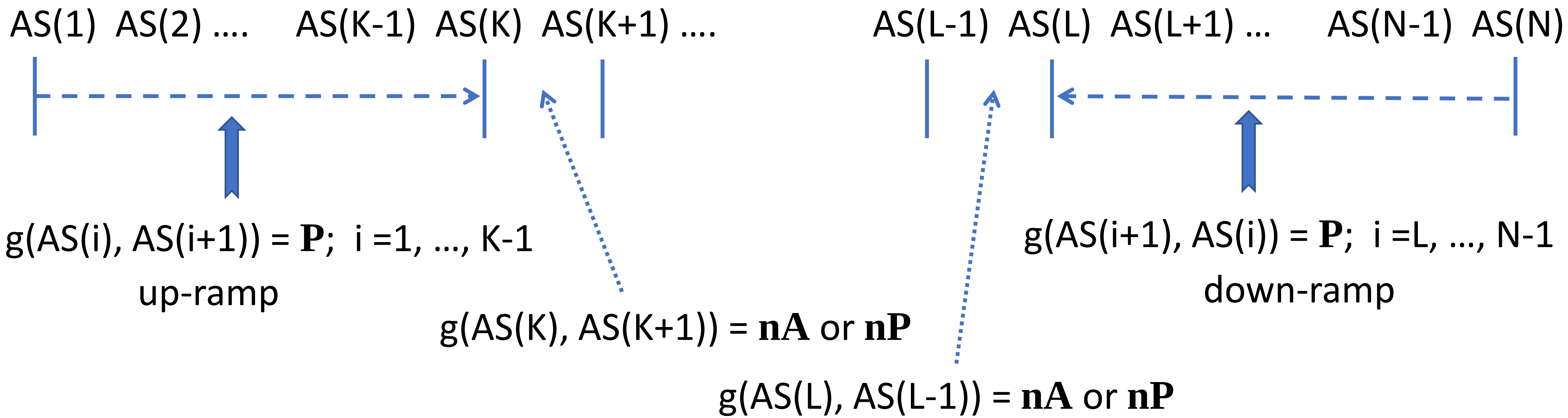
ASPAs:
AS(1), AS(2)
AS(2), AS(3)
AS(5), AS(4)
AS(6), AS(5)
AS(7), AS(6)



ASPA of AS(3) does not exist or it does not include AS(4)
ASPA of AS(4) does not exist or it does not include AS(3)

- The AS path is Valid with/without the **nP** or **nA** hop in the middle
- AS path is trivially Valid if the AS path length is 1 or 2 (no ASPA needed)

(K, L) representation of downstream AS path

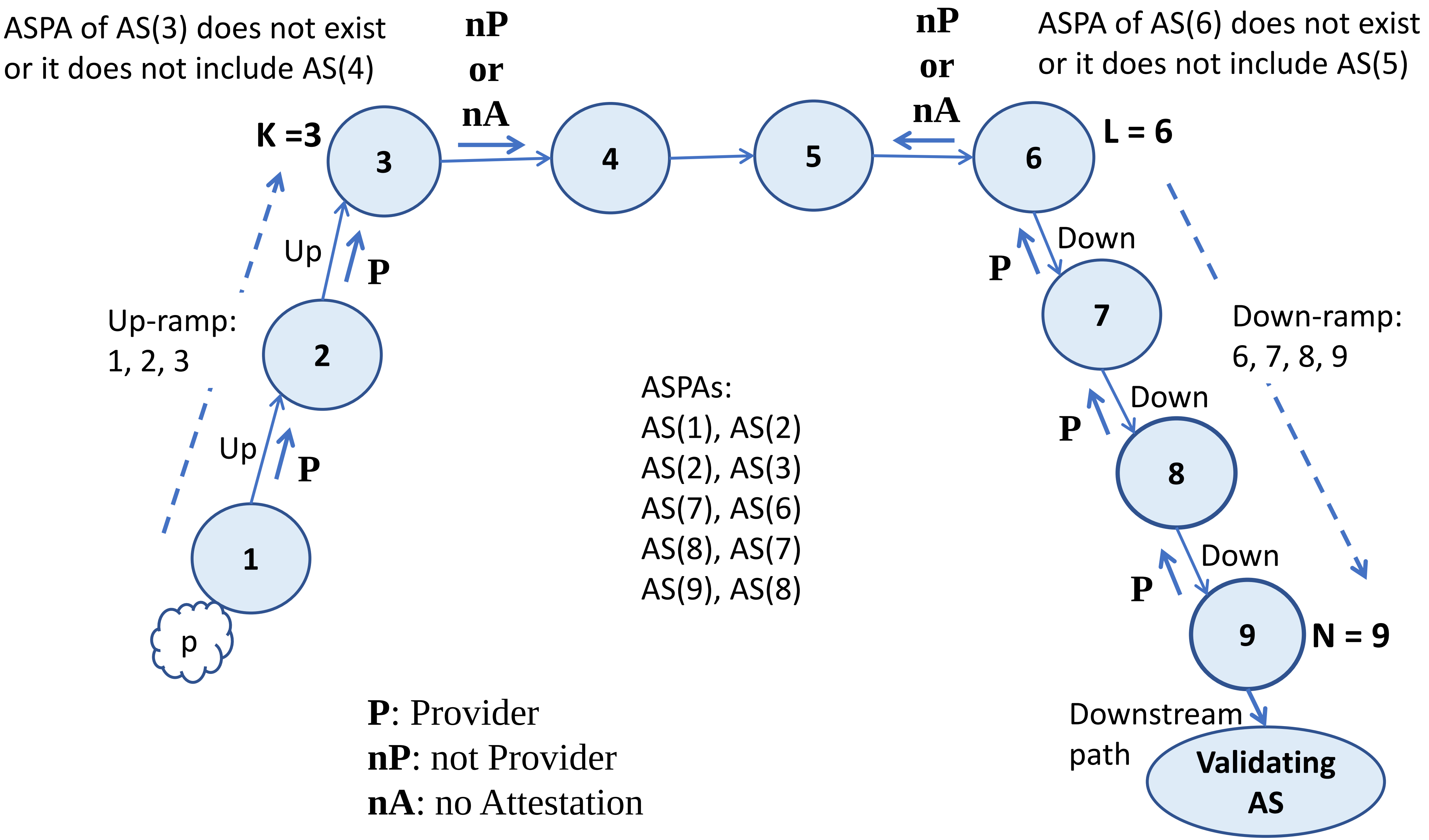


g is the ASPA hop check function as defined on slide 3.

- If $L - K \leq 1$, then the AS path is Valid

ASPA hop check:
P: Provider
nP: not Provider
nA: no Attestation

(K, L) representation of downstream AS path



When L < K

- L < K means that the up-ramp and down-ramp overlap
  - ➢ Obvious that the AS path is Valid
- Happens when there are siblings in the middle of the path
  - ➢ Siblings: neighbors AS(i) and AS(j) each attest the other as provider
- During computation, the down-ramp determination can be halted when the top of the down-ramp touches the top of the up-ramp:
  - ➢ AS(1) to AS(K) is regarded as the up-ramp
  - ➢ AS(K) to AS(N) is regarded as the down-ramp

Theorems that help design the algorithm

Theorem 1: The downstream AS path is “Valid” if and only if L-K $\leq$ 1. If L-K $\geq$ 2, then the AS path can be “Unknown” or “Invalid”, but never “Valid”.

Theorem 2: For L-K $\geq$ 2, the validity of the whole AS path is the same as that of the partial path AS(K), AS(K+1), …., AS(L-1), AS(L). The partial path can only be either Invalid or Unknown. It is Invalid if there exist u and v (u and v in the range from K to L-1) such that u $<$ v and g(AS(u), AS(u+1)) = $\mathbf{nP}$ and g(AS(v+1), AS(v)) = $\mathbf{nP}$. Otherwise, the partial path is Unknown.

Function g is defined on slide 3.

Proofs exist; discussed in the next slide and backup slides.

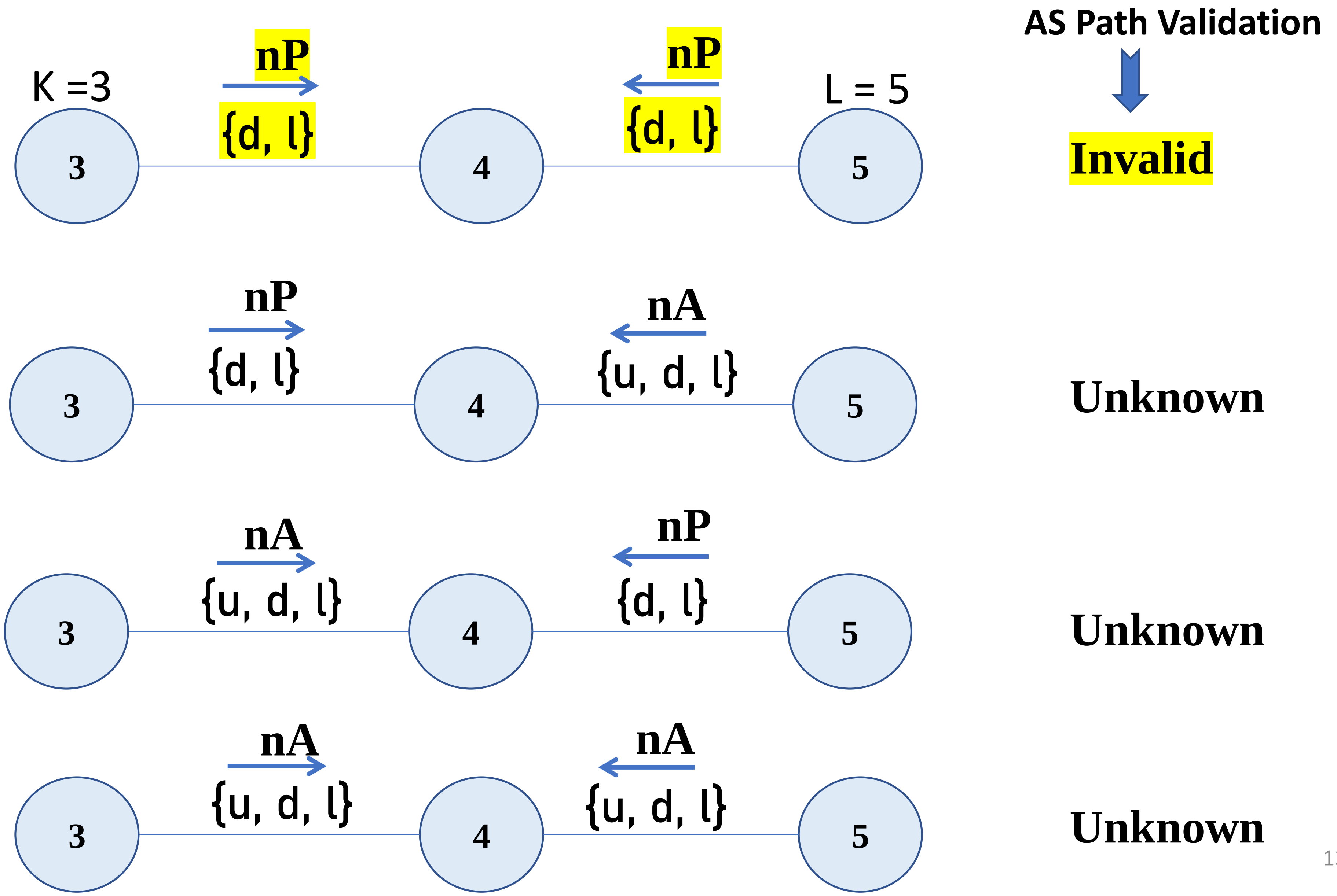
For L-K ≥ 2, only Invalid or Unknown are possible

Illustration for L-K = 2

ASPA hop check:
P: Provider
nP: not Provider
nA: no Attestation

AS-AS peering:
u = Up
d = Down
l = Lateral

Arrows indicate direction of ASPA hop check



Algorithm for ASPA based downstream AS path validation

Crisp Description

Formulate the AS path using the (K, L) representation*.
If the AS path length $N \leq 2$, then the path is Valid and the procedure halts.
If $L\text{-}K \leq 1$, then the AS path is Valid and the procedure halts.
(Note: For $L\text{-}K \geq 2$, to determine whether the AS path is Invalid or Unknown,
we only need to focus on the portion of the path from AS(K) to AS(L).)
Consider the partial path represented by AS(K), AS(K+1), …, AS(L-1), AS(L).
For $L\text{-}K \geq 2$, if there exist u and v in the range from K to L-1 such that $u < v$ and
g(AS(u), A(u+1), AFI) = **nP**, and
g(AS(v+1), A(v), AFI) = **nP**,
then the AS path is Invalid and the procedure halts.
Else, the AS path is Unknown.

* Collapsing mutual transit ASes (siblings) and keeping only one of them in the AS path (when they each have ASPA attesting the other as provider) is not necessary.

Algorithm for ASPA based downstream AS path validation

Implementation procedure

1. If there is an AS_SET present in the AS path, then set AS_SET_FLAG = 1, else AS_SET_FLAG = 0.
2. Collapse prepends in the AS_SEQUENCE in the AS path so that each AS number (ASN) in the path is unique. Call this path (after collapsing the prepends) as the AS path for this algorithm.
3. If the AS path in step 2 is empty, then go to step 11*.
4. Let the AS path be represented as AS(1), AS(2), …, AS(N-1), AS(N), where N is the AS path length and AS(N) is the most recently added AS in the AS path and neighbor to the receiving/validating AS.
5. If $N \leq 2$, then the update is "Valid" and go to step 11.
6. At this step, N ≥ 3. Evaluate sequentially starting from i = 1 (1 ≤ i ≤ N-2) and determine the largest i (= i_max) such that g(AS(i), AS(i+1), AFI) = $\mathbf{P}$ for each i ≤ i_max. If there is no such i_max, then set i_max = 0. Let K = i_max + 1. If K = N-1, then the AS path is "Valid" and go to step 11.
7. Evaluate sequentially starting from j = 1 (1 ≤ j ≤ N-K-1) and determine the largest j (= j_max) for which g(AS(N-j+1), AS(N-j), AFI) = $\mathbf{P}$ for each j ≤ j_max. If there is no such j_max, then set j_max = 0. Let L = N - j_max.
8. If L-K ≤ 1, then the AS path is "Valid" and go to step 11.
9. At this step, L-K ≥ 2. Record the lowest value of i (K ≤ i ≤ L-2) for which g(AS(i), AS(i+1), AFI) = $\mathbf{nP}$ and set u equal to that lowest value. If no such value for u exists, then go to step 10. Else, find a value of j (u+1 ≤ j ≤ L-1) for which g(AS(j+1), AS(j), AFI)) = $\mathbf{nP}$. If such j is found, then the AS path is "Invalid" and the procedure halts.
10. If AS_SET_FLAG = 0, then the Update is "Unknown". The procedure halts.
11. If AS_SET_FLAG = 1, then the Update is "Unverifiable". The procedure halts.

* Totally empty AS_PATH (no AS_SEQUENCE, no AS_SET) would be an error in eBGP.

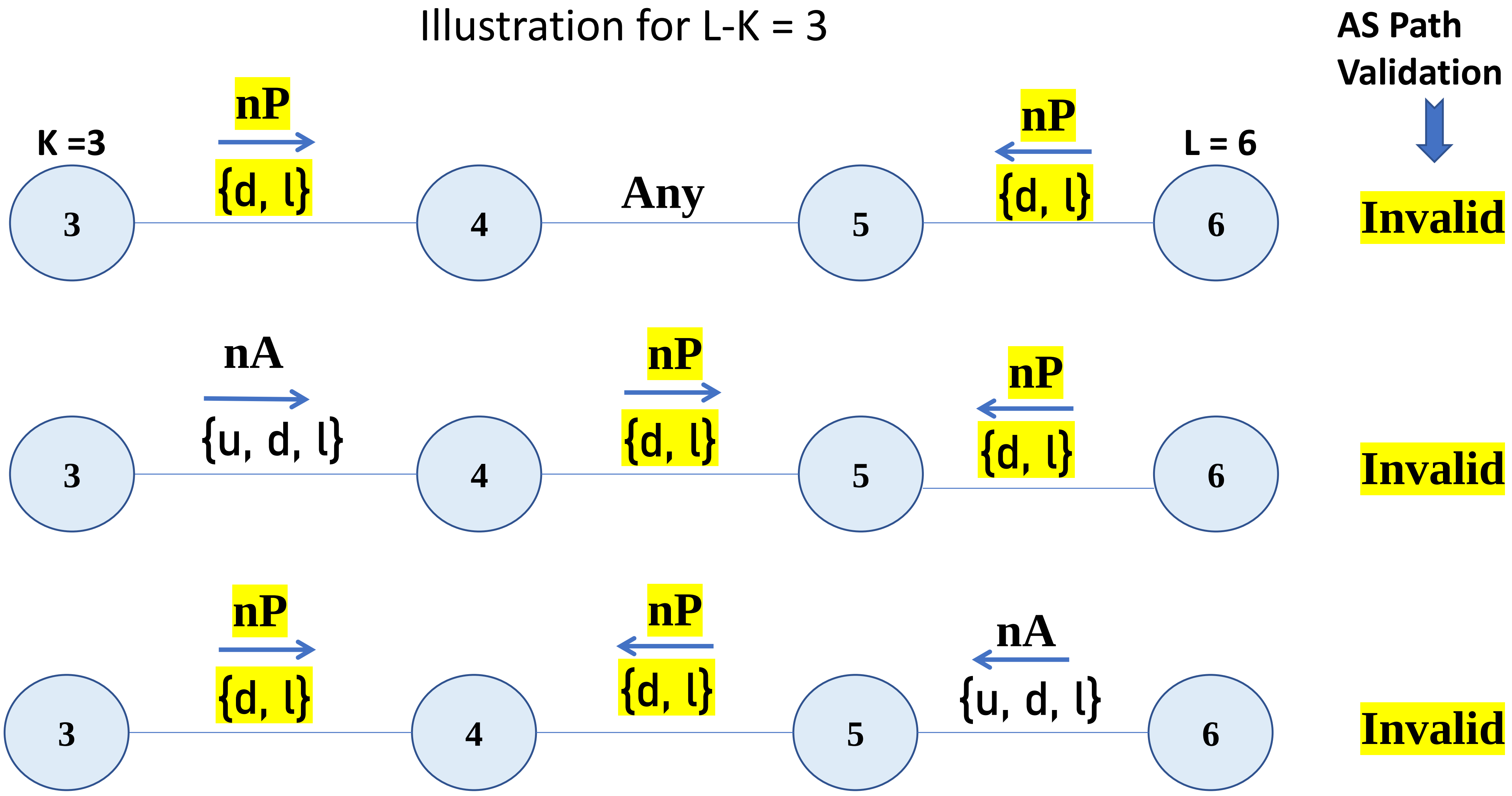
Backup slides

Proofs of the Theorems

Proof: For L-K $\geq$ 2, only Invalid or Unknown are possible

ASPA hop check:
P: Provider
nP: not Provider
nA: no Attestation

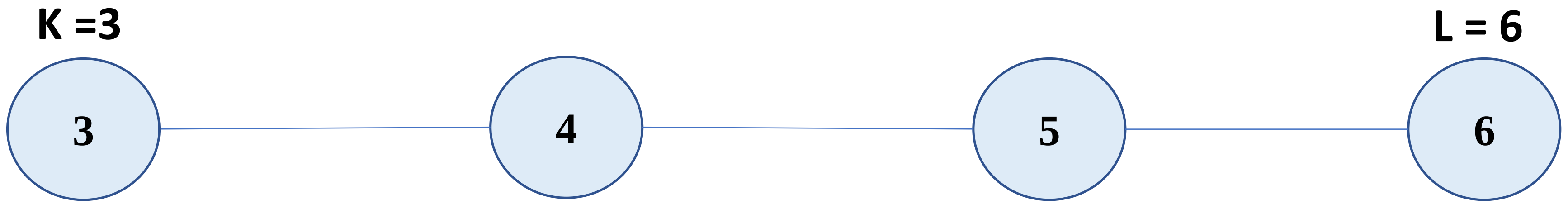
AS-AS peering:
u = Up
d = Down
l = Lateral



Arrows indicate direction of ASPA hop check

Proof: For L-K ≥ 2, only Invalid or Unknown are possible

Illustration for L-K = 3



ASPA hop check:
P: Provider
nP: not Provider
nA: no Attestation

AS-AS peering:
u = Up
d = Down
l = Lateral

→ ← Arrows indicate direction of ASPA hop check

Hop 3-4	**Hop 4-5**	**Hop 5-6**	**AS path**
→ **nP** {d, l}	Any: **P, nP, or nA**	← **nP** {d, l}	Invalid
→ **nP** {d, l}	← **nP** {d, l}	← **nA** {u, d, l}	Invalid
→ **nP** {d, l}	← **nA** {u, d, l}	← **nA** {u, d, l}	Unknown
→ **nP** {d, l}	← **P** {u}	← **nA** {u, d, l}	Unknown
→ **nA** {u, d, l}	→ **nP** {d, l}	← **nP** {d, l}	Invalid
→ **nA** {u, d, l}	→ **nP** {d, l}	← **nA** {u, d, l}	Unknown
→ **nA** {u, d, l}	→ **nA** {u, d, l}	← **nP** {d, l}	Unknown
→ **nA** {u, d, l}	→ **nA** {u, d, l}	← **nA** {u, d, l}	Unknown
→ **nA** {u, d, l}	→ **P** {u}	← **nP** {d, l}	Unknown
→ **nA** {u, d, l}	→ **P** {u}	← **nA** {u, d, l}	Unknown

Proof: For L-K ≥ 2, only Invalid or Unknown are possible

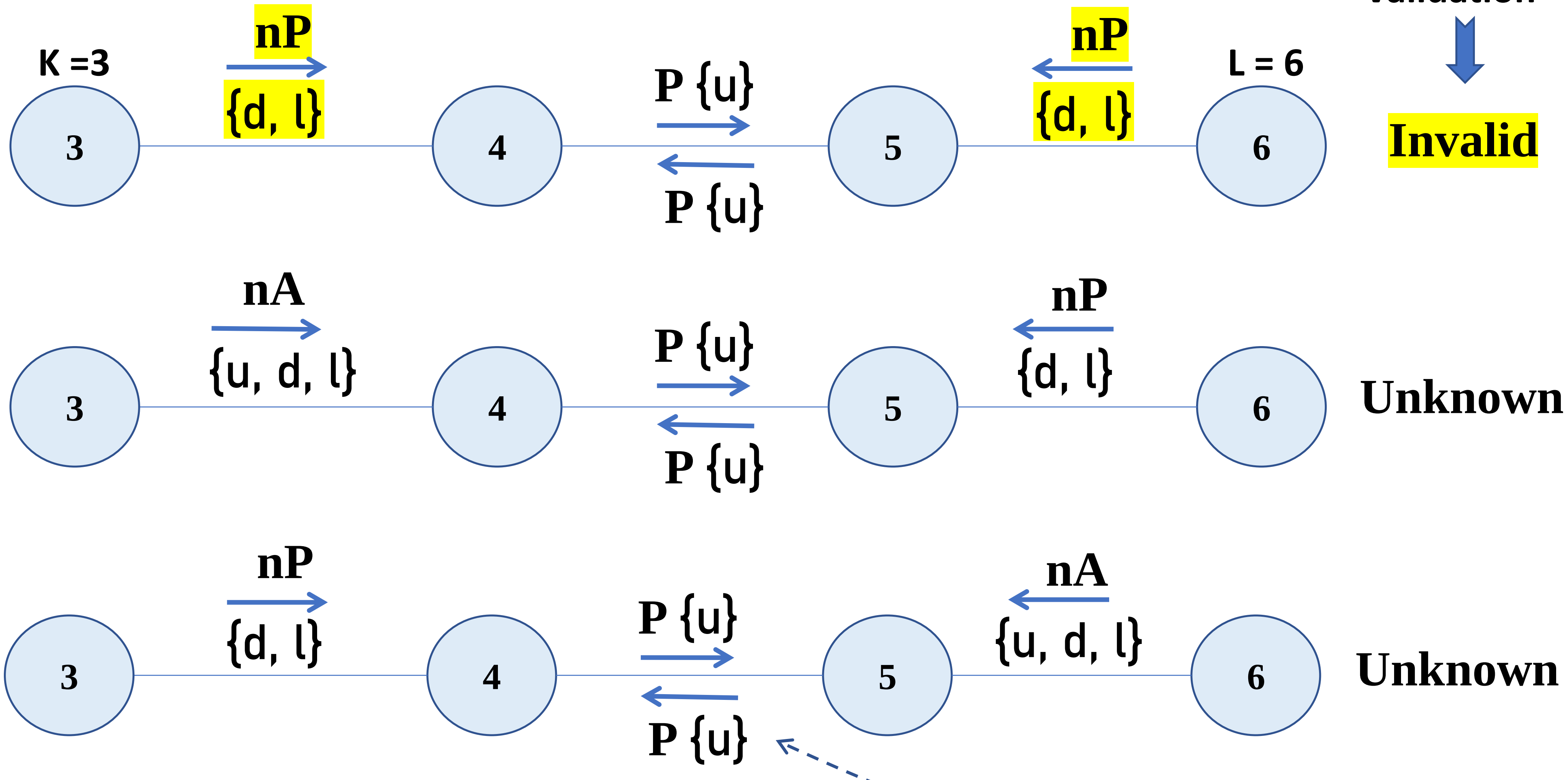
Theorems stay true in the presence of sibling hops

Illustration for L-K = 3

AS Path Validation

ASPA hop check:
P: Provider
nP: not Provider
nA: no Attestation

AS-AS peering:
u = Up
d = Down
l = Lateral



Arrows indicate direction of ASPA hop check

AS(4) and AS(5) are siblings

Proof of Theorem 2

Theorem 2 has been shown to be correct by enumeration for L-K = 2 and L-K = 3 (see slides 13, 17, 18, 19). Now the proof can be completed by the method of induction. It can be shown that if the assertion is true for L-K = n, then it also true for L-K = n+1, for any value of n. This will be described in a paper.