Bottleneck Structure Graphs in Multidomain Networks: Introduction and Requirements for ALTO

Jordi Ros-Giralt, Sruthi Yellamraju, Qin Wu, Richard Yang, Luis Contreras, Kai Gao, Jensen Zhang

I-Draft: draft-giraltyellamraju-alto-bsg-multidomain

https://datatracker.ietf.org/doc/draft-giraltyellamraju-alto-bsg-multidomain/

IETF Plenary 115

ALTO WG Session

11/11/2022

Table of Contents

- Brief Introduction to Bottleneck Structures
- Computing bottleneck structures under partial information
- Requirements discussion

Brief Introduction to Bottleneck Structures

Framework and Implementation Details in the Following I-Drafts and Papers

- [1] IETF Draft: “Supporting Bottleneck Structure Graphs in ALTO: Use Cases and Requirements”, https://datatracker.ietf.org/doc/draft-giraltyellamraju-alto-bsg-requirements/
- [2] IETF Draft: “Bottleneck Structure Graphs in Multidomain Networks: Introduction and Requirements for ALTO”, https://datatracker.ietf.org/doc/draft-giraltyellamraju-alto-bsg-multidomain/
- [3] "On the Bottleneck Structure of Congestion-Controlled Networks," ACM SIGMETRICS, Boston, June 2020 [https://bit.ly/3Urng9M].
- [4] "Designing Data Center Networks Using Bottleneck Structures," accepted for publication at ACM SIGCOMM 2021 [https://bit.ly/3TaJpZ5].
- [5] "A Quantitative Theory of Bottleneck Structures for Data Networks", Qualcomm Technologies, Inc. Technical Report, 2022 [https://bit.ly/3DG4u7U].

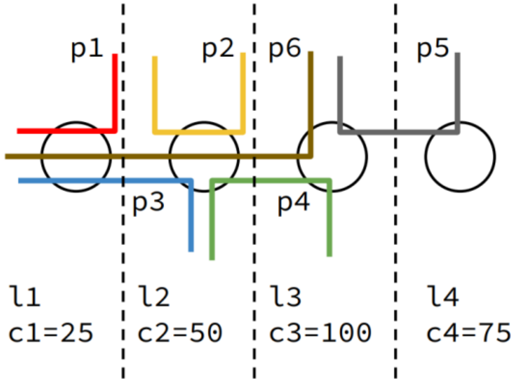
See also the IETF 115 PANRG sessions from yesterday (11/10/2022) for complete details:

10:05	20m	Computing Path Metrics Using Bottleneck Structure Graphs, draft-giraltyellamraju-alto-bsg-requirements	Jordi Ros-Giralt, Qin Wu
10:25	20m	Computing Bottleneck Structure Graphs in Multi-Domain Networks, draft-giraltyellamraju-alto-bsg-multidomain	Jordi Ros-Giralt, Qin Wu

Computing Bottleneck Structures Under Partial Information

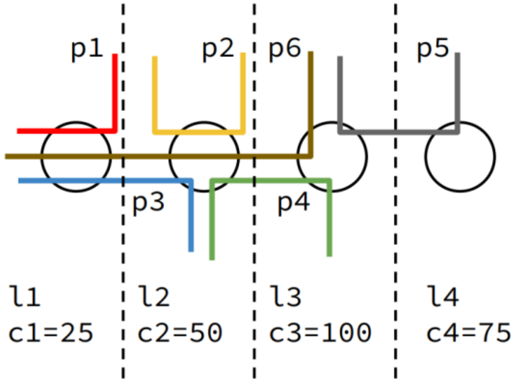
Bottleneck Structures Under Partial Information: Problem Statement

Network with a single AS:

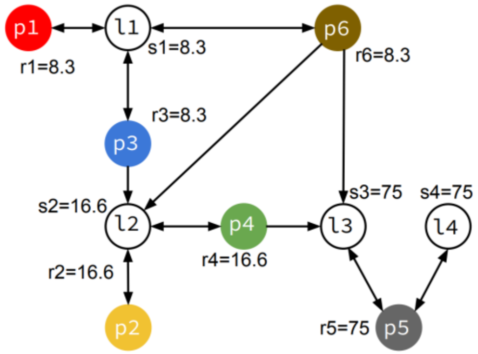


Bottleneck Structures Under Partial Information: Problem Statement

Network with a single AS:

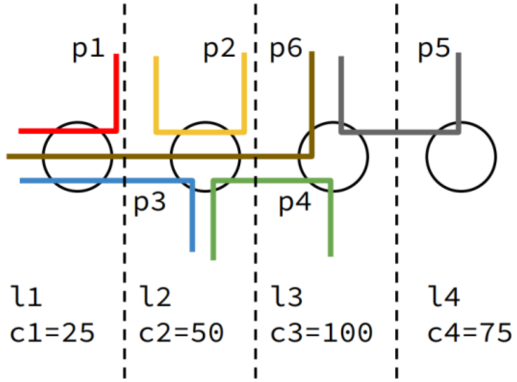


Bottleneck Structure (Path Gradient Graph):

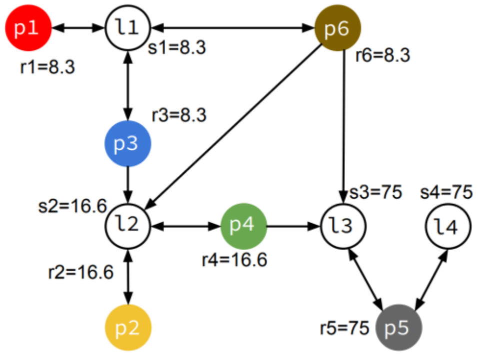


Bottleneck Structures Under Partial Information: Problem Statement

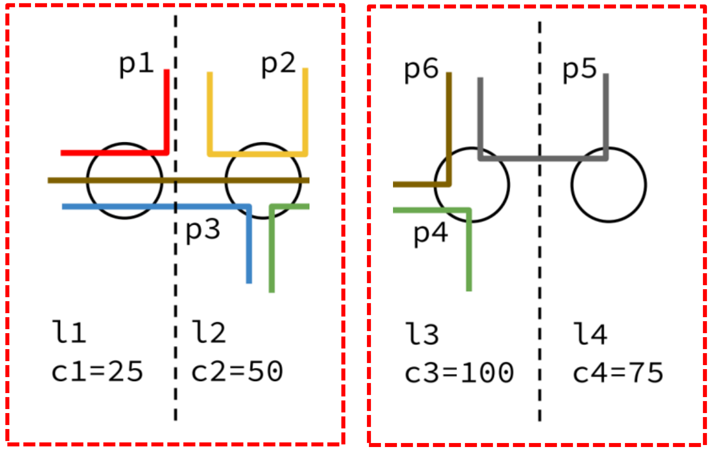
Network with a single AS:



Bottleneck Structure (Path Gradient Graph):



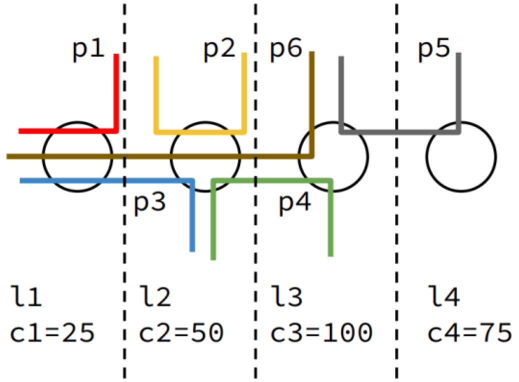
Network with multiple ASs:



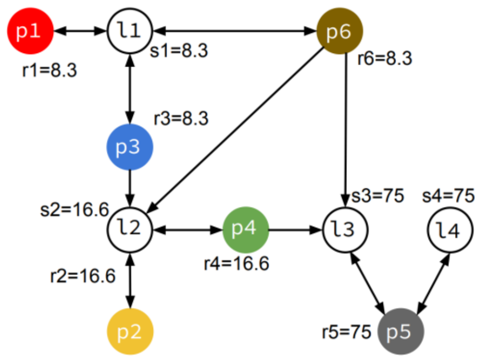
AS1 AS2

Bottleneck Structures Under Partial Information: Problem Statement

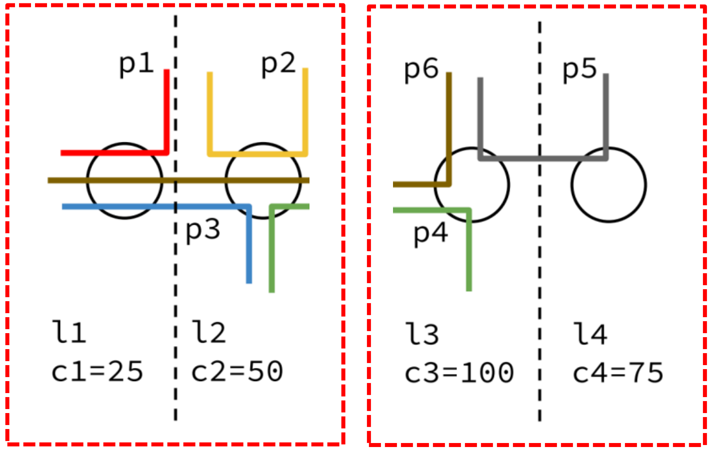
Network with a single AS:



Bottleneck Structure (Path Gradient Graph):

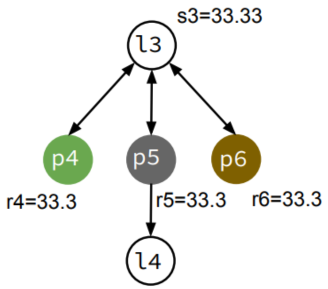


Network with multiple ASs:



AS1 AS2

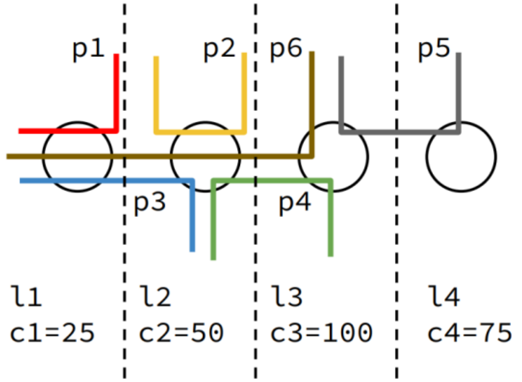
Bottleneck Structure of AS2:



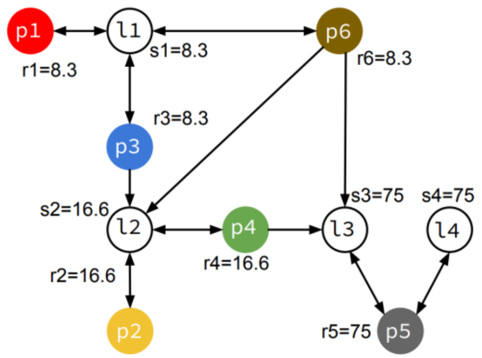
Incorrect substructure

Bottleneck Structures Under Partial Information: Problem Statement

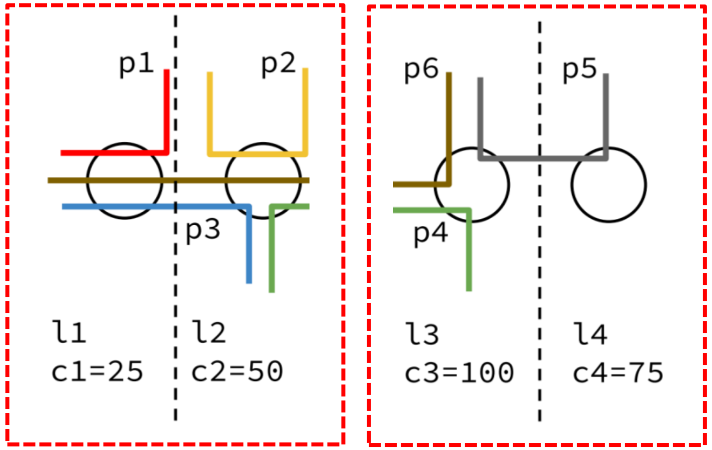
Network with a single AS:



Bottleneck Structure (Path Gradient Graph):



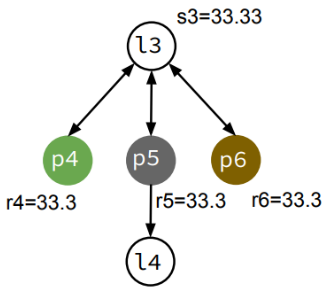
Network with multiple ASs:



AS1

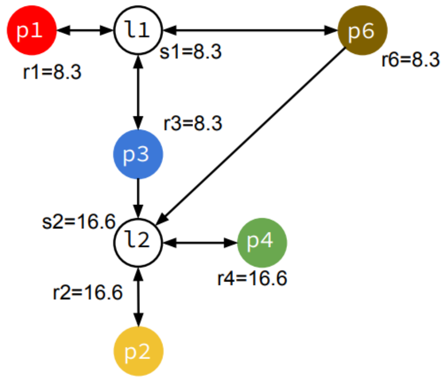
AS2

Bottleneck Structure of AS2:



Incorrect substructure

Bottleneck Structure of AS1:



Correct substructure (got lucky)

Proposed Distributed Border Protocol

- **Convergence** (key idea): Sharing one metric per path is enough to ensure convergence to the correct bottleneck substructures.
- **Scalability**: Focuses on building the path gradient graph (see draft-giraltyellamraju-alto-bsg-requirements), requires only per-path state (not per-flow).
- **Privacy**: Does not require sharing sensitive flow information. Only one metric (scalar) per path is shared with the neighbor ASs.

Notation	Description
A	The set of autonomous systems (ASs).
A_i	An AS in A, for i = 1, ..., |A|.
P(A_i) = {p_1, ..., p_|P(A_i)|}	The set of active paths found in A_i. These are paths for which there exist 0traffic flowing through them.
L(A_i) = {l_1, ..., l_|L(A_i)|}	The set of active links found in A_i. These are links for which there exists traffic flowing through them.
B	The global bottleneck structure graph. The form of bottleneck structure used by the distributed algorithm introduced in this document is the Path Gradient Graph [I-D.draft-giraltyellamraju-alto-bsg-requirements].
B.BW(p)	The bandwidth available to path p according to the global bottleneck structure. This is always the globally correct available bandwidth for path p.
B(A_i)	The bottleneck substructure of A_i, corresponding to the subgraph of B that includes (1) the vertices corresponding to the paths in P(A_i), (2) the vertices corresponding to the links in L(A_i) and (3) all the edges in B that connect them. If a path p in P(A_i) is bottlenecked at a link not in L(A_i), then B(P, L) includes a virtual link v with capacity equal to B.BW(p) and a directed edge from v to p.
B(A_i).BW(p)	The bandwidth available to path p according to the bottleneck substructure of A_i. This value is equal to B.BW(p) when the distributed algorithm terminates.
PL(A_i)	A dictionary mapping every path in P(A_i) with the subset of links in L(A_i) that it traverses. Note that a path p can traverse one or more links not in L(A_i). This reflects the notion of partial information inherent to multi-domain networking environments. That is, A_i may not know all the links traversed by its active paths; in particular, it only knows the subset of links that are in A_i.
C(A_i)	A dictionary mapping each link in A_i with its capacity (in bps).
N(A_i)	The set of ASs that are neighbors of A_i.
PM(A_i)(p)	The current bandwidth available to path p as computed by A_i. This is also known as the path metric of p according to A_i.

Table 1: Conventions and definitions used in the description of the distributed protocol.

Description of the Distributed Protocol

The algorithm run by each autonomous system AS_i, 1 <= i <= |A|, consists of two independently executed events as follows:

Event: TIMER

```
- Every s seconds, perform the following tasks:

    1. B(A_i) = COMPUTE_BOTTLENECK_SUBSTRUCTURE(L(A_i), PL(A_i), C(A_i), PM(A_i));

    2. PM(A_i)(p) = B(A_i).BW(p), for all p in P(A_i);

    3. For all A_j in N(A_i):

        3.1 Send to A_j a PATH_METRIC_ANNOUNCEMENT message including (AS_i, PM(A_i));
```

Event: PATH_METRIC_EXCHANGE

```
- Upon receiving a PATH_METRIC_ANNOUNCEMENT from AS_j carrying (AS_j, PM(A_j)):

    1. PM(A_i)(p) = min{PM(A_i)(p), PM(A_j)(p)}, for all p in P(A_i) and p in P(A_j);
```

Description of the Distributed Protocol

Procedure: COMPUTE_BOTTLENECK_SUBSTRUCTURE(L, PL, C, PM):

```
1. i = 0; L_0 = L; PL_0 = PL;
2. While True:
        2.1. B_i = COMPUTE_BOTTLENECK_STRUCTURE(L_i, PL_i, C);
        2.2. If B_i.BW(p) == PM(p) for all path p in PL_i:
                2.2.1. Break;
        2.3. For all path p in PL_i such that B_i.BW(p) > PM(p):
                2.3.1. If PL_i[p] has no virtual link:
                        2.3.1.1. Add a new virtual link v to the set of links PL_i[p];
                        2.3.1.2. Add virtual link v to the set L_i;
                2.3.2.  Set C(v) = PM(p);
        2.2. i = i + 1;
        2.5. L_i = L_{i-1};
        2.6. PL_i = PL_{i-1};
3. Return B_i;
```

Termination and Convergence Conditions

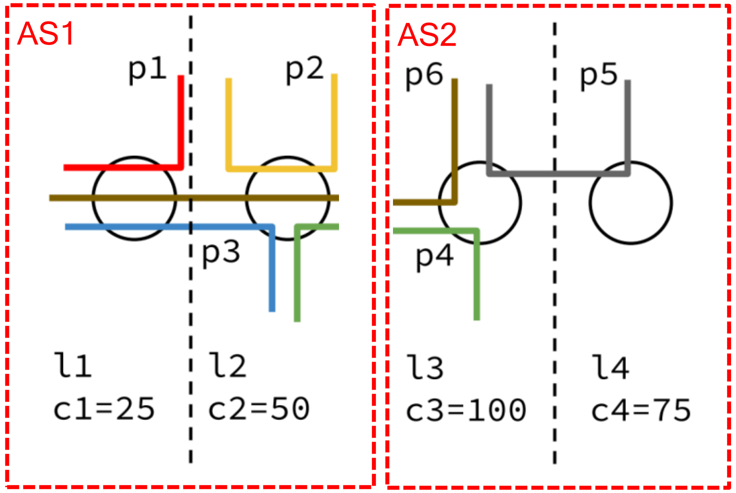
Termination Condition. When the output of an autonomous system's local computation of the bottleneck structure is in agreement with the Path Metric Dictionary it maintains in coordination with its neighbors, the algorithm terminates:

```
B_i.BW(p) == PM(p) for all path p in PL_i
```

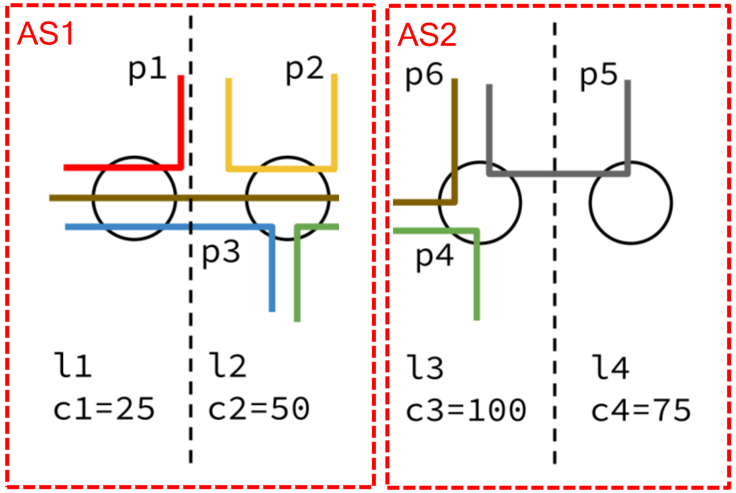
Convergence Condition. Upon termination, the Path Metric Dictionaries of all the autonomous systems are in agreement with each other:

```
PM(A_i)(p) == PM(A_j)(p) for all p in A_i, p in A_j, A_i in A and A_j in A
```

Example: Global Convergence to the Correct Bottleneck Substructures



Example: Global Convergence to the Correct Bottleneck Substructures

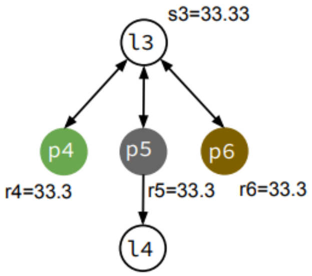


AS2

Iteration 1:

PMD(AS1)

p4:	33.3
p5	33.3
p6	33.3



Sharing of PMDs using a PATH_METRIC_ANNOUNCEMENT message between neighbour ASs

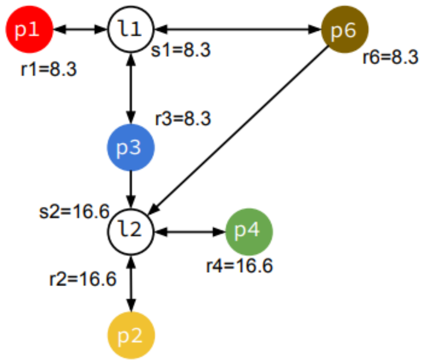
AS1

Bottleneck Substructure for AS1:

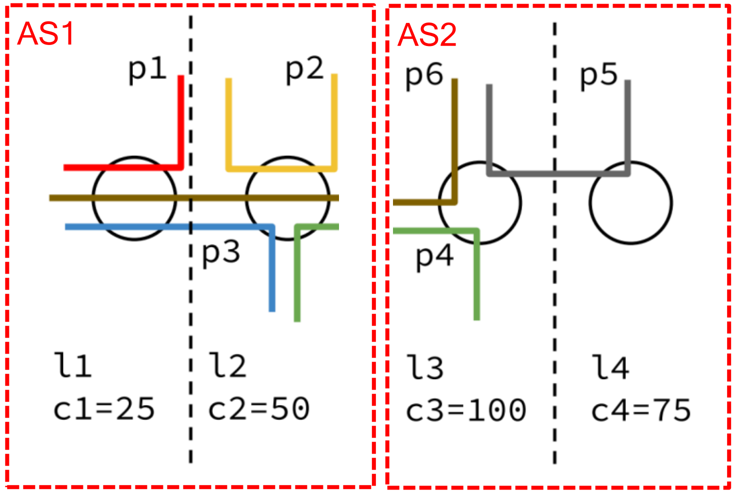
Iteration 1:

PMD(AS1)

p1	8.3
p2	16.6
p3	8.3
p4	16.6
p6	8.3



Example: Global Convergence to the Correct Bottleneck Substructures

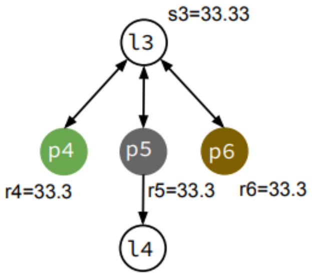


AS2

Iteration 1:

PMD(AS1)

p4:	33.3
p5	33.3
p6	33.3

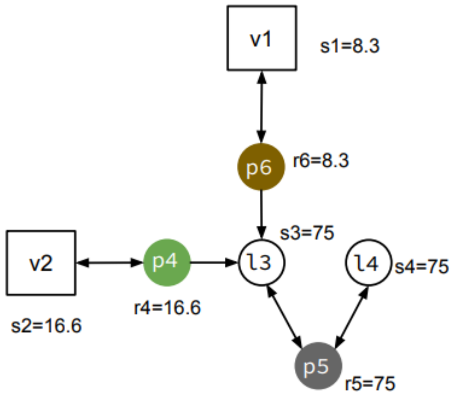


Iteration 2:

PMD(AS2)

p4	16.6
p5	75
p6	8.3

Bottleneck Substructure for AS2:



Sharing of PMDs using a PATH_METRIC_ANNOUNCEMENT message between neighbour ASs

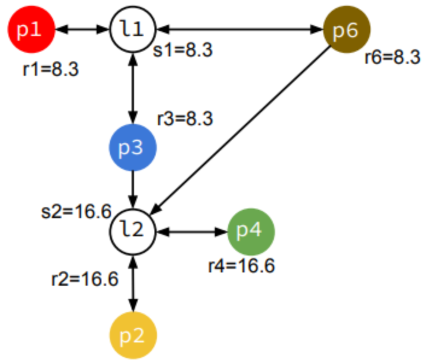
AS1

Iteration 1:

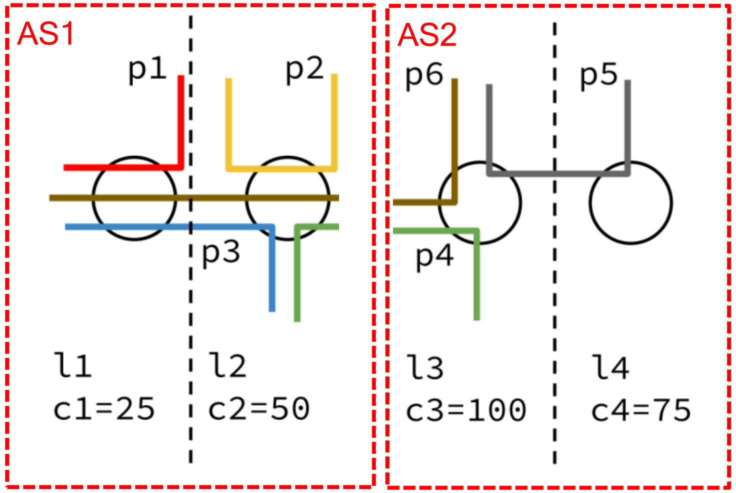
PMD(AS1)

p1	8.3
p2	16.6
p3	8.3
p4	16.6
p6	8.3

Bottleneck Substructure for AS1:



Example: Global Convergence to the Correct Bottleneck Substructures



Convergence Condition

```
PM(A_i)(p) = PM(A_j)(p)
for all p in A_i, p in A_j, A_i in A and A_j in A
```

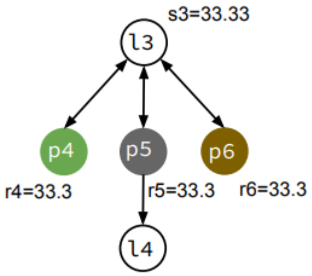
p	PM(A1)(p)	PM(A2)(p)
p1	8.3	--
p2	16.6	--
p3	8.3	--
p4	16.6	16.6
p5	--	75
p6	8.3	8.3

AS2

Iteration 1:

PMD(AS1)

p4:	33.3
p5	33.3
p6	33.3

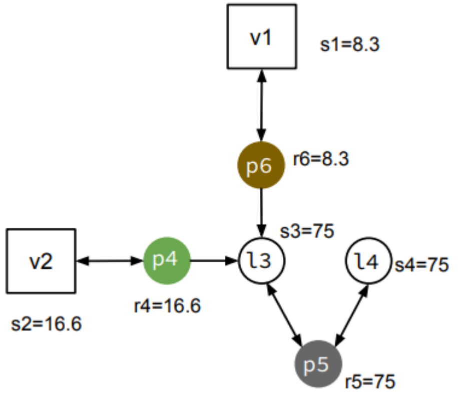


Iteration 2:

PMD(AS2)

p4	16.6
p5	75
p6	8.3

Bottleneck Substructure for AS2:



Sharing of PMDs using a PATH_METRIC_ANNOUNCEMENT message between neighbour ASs

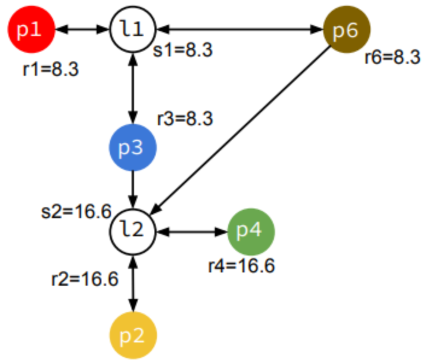
AS1

Bottleneck Substructure for AS1:

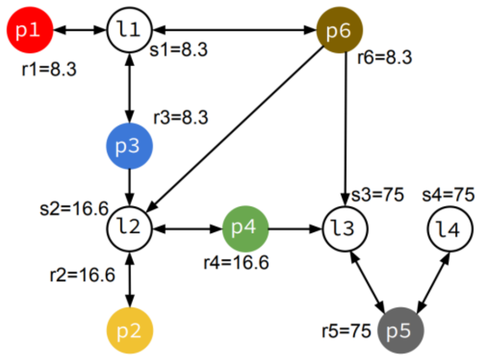
Iteration 1:

PMD(AS1)

p1	8.3
p2	16.6
p3	8.3
p4	16.6
p6	8.3

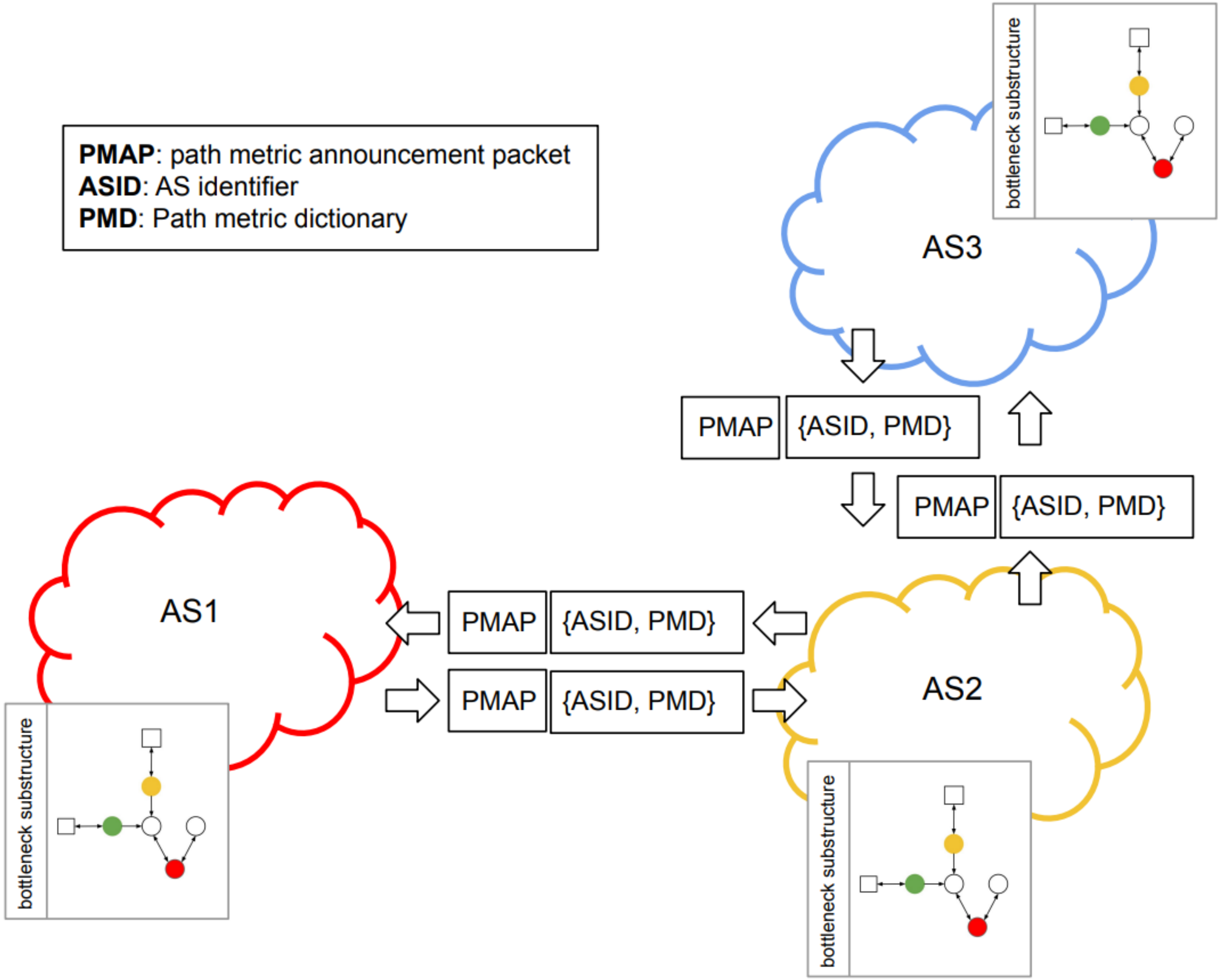


Global Bottleneck Structure:



Example: Global Convergence to the Correct Bottleneck Substructures

PMAP: path metric announcement packet
ASID: AS identifier
PMD: Path metric dictionary



Requirements discussion

Requirements Discussion

To implement the proposed distributed protocol using ALTO, two broad requirements are necessary:

- Requirement 1: The capability for each ALTO server to compute bottleneck substructures of its own AS.
- Requirement 2: The capability for each ALTO server to communicate with its neighboring ASs.

4.1. Requirement 1: Computation of Bottleneck Substructures

The requirements for an ALTO server to compute the bottleneck substructure of its associated AS are the same as the requirements to compute the bottleneck structure in the case the network consists of a single autonomous system. These requirements are discussed in the Requirements Section of [I-D.draft-giraltyellamraju-alto-bsg-requirements]. Refer to this document for further details.

4.2. Requirement 2: Communication Between Neighboring ASs

The TIMER event executed by each ALTO server needs to periodically transmit a PATH_METRIC_ANNOUNCEMENT message to its neighboring ASs. This leads to the following requirement:¶

- Requirement 2.1: ALTO servers managing neighboring ASs need to be reachable to each other.
- Requirement 2.2: The sharing of algorithmic state between ALTO servers requires extending the base ALTO protocol to support server-to-server communication semantics.

Discussion Q&A

Thank you