Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 10, 2015

S. Randriamasy
W. Roome
Alcatel-Lucent
N. Schwan
Thales Deutschland

March 9, 2015

Multi-Cost ALTO
draft-randriamasy-alto-multi-cost-10

Abstract

IETF is designing a new service called ALTO (Application Layer traffic Optimization) that includes a "Network Map Service", an "Endpoint Cost Service" and an "Endpoint (EP) Ranking Service" and thus incentives for application clients to connect to ISP preferred Endpoints. These services provide a view of the Network Provider (NP) topology to overlay clients.

The present draft proposes a simple way to extend the information provided by the current ALTO protocol in two ways. First, including information on multiple Cost Types in a single ALTO transaction provides a better mapping of the Selected Endpoints to needs of the growing diversity of Content and Resources Networking Applications and to the network conditions. Second, one ALTO query and response exchange on N Cost Types is faster and more efficient than N single cost transactions. All this also helps producing a faster and more robust choice when multiple Endpoints need to be selected. Last, the draft proposes to enrich the filtering capabilities by allowing constraints involving several metrics combined by several types of logical operators. This allows the applications to set finer requirements and above all to include compromises on those requirements.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

IETF has designed a new service called ALTO that provides guidance to overlay applications, which have to select one or several hosts from a set of candidates that are able to provide a desired resource. This guidance is based on parameters that affect performance and efficiency of the data transmission between the hosts, e.g., the topological distance. The purpose of ALTO is to improve Quality of Experience (QoE) in the application while reducing resource consumption in the underlying network infrastructure. The ALTO protocol conveys the Internet View from the perspective of a Provider Network region that spans from a region to one or more Autonomous System (AS). Together with this Network Map, it provides the Provider determined Cost Map between locations of the Network Map. Last, it provides the Ranking of Endpoints w.r.t. their routing cost.

Current ALTO Costs and their modes provide values that are seen to be stable over a longer period of time, such as hopcount and administrative routing cost to reflect ISP routing preferences. Recently, new use cases have extended the usage scope of ALTO to Content Delivery Networks, Data centers and applications that need additional information to select their Endpoints or handle their PIDs.

Thus a multitude of new Cost Types that better reflect the requirements of these applications are expected to be specified, in particular cost values that change more frequently than previously assumed.

The ALTO protocol [RFC7285] restricts ALTO Cost Maps and Endpoint Cost services to only one Cost Type and Cost Mode per ALTO request. To retrieve information for several Cost Types, an ALTO client must send several separate requests to the server.

It would be far more efficient, in terms of RTT, traffic, and processing load on the ALTO client and server, to get all costs with a single query/response transaction. Vector costs provide a robust and natural input to multi-variate path computation as well as robust multi-variate selection of multiple Endpoints. In particular, one Cost Map reporting on N Cost Types is less bulky than N Cost Maps containing one Cost Type each. This is valuable for both the storage of these maps and their transmission. Additionally, for many emerging applications that need information on several Cost Types, having them gathered in one map will save time.

Along with multi-cost values queries, the filtering capabilities need to be extended to allow constraints on multiple metrics. The base protocol allows optional constraints in the input parameters to a request for a Filtered Cost Map or the Endpoint Cost Service. The ’constraints’ member is an AND-combination of expressions that all apply to the (single) requested Cost Type. It is therefore necessary to allow constraints on multiple metrics. Beyond that, applications that are sensitive to several metrics and struggle with complicated network conditions may need to arbitrate between conflicting objectives such as routing cost and network performance. To address this issue, this draft proposes to extend the base protocol by both allowing to combine constraints on multiple metrics and relating these constraints with a logical ’AND’ and a logical ’OR’. This allows an application to make compromises such as: "select solutions with either (moderate ’hopcount’ AND high ’routingcost’) OR (higher ’hopcount’ AND moderate ’routingcost’)".

This draft is organized as follows: section 3 exposes use cases motivating the introduction of new Cost Types and why multi-cost transactions are useful. Section 4 identifies the core ALTO protocol extensions that are required or recommended to support requests and responses on multiple Cost Types in one single transaction. Section 5 specifies the extended constraints on mutli-cost values. Section 6 specifies the protocol extensions for Multi-Cost ALTO transactions and provides examples.

2. Application Scope And Terminology

This draft generalizes the case of a P2P client to include the case of a CDN client, a client of an application running on a virtual server, a GRID application client and any Client having the choice in several connection points for data or resource exchange. To do so, it uses the term "Application Client" (AC).

This draft focuses on the use case where the ALTO client is embedded in the Application Client or in some Application Endpoint tracker in the network, such as a P2P tracker, a CDN request router or a cloud computing orchestration system implemented in a logically centralized management system.

It is assumed that Applications likely to use the ALTO service have a choice in connection endpoints as it is the case for most of them. The ALTO service is managed by the Network Provider (NP) and reflects its preferences for the choice of endpoints. The NP defines in particular the network map, the routing cost among Network Locations, the cost types used to reflect it, and which ALTO services are available at a given ALTO server.

This draft uses terms defined as follows:

- Endpoint (EP): can be a Peer, a CDN storage location, a physical server involved in a virtual server-supported application, a Party in a resource sharing swarm such as a computation Grid or an online multi-party game.

- Endpoint Discovery (EP Discovery) : this term covers the different types of processes used to discover the eligible endpoints.

- Network Service Provider (NSP): includes both ISPs, who provide means to transport the data, and Content Delivery Networks (CDNs) who care for the dissemination, persistent storage and possibly identification of the best/closest content copy.

- ALTO transaction: a request/response exchange between an ALTO Client and an ALTO Server.

- Application Client (AC): this term generalizes the case of a P2P client to include the case of a CDN client, a client of an application running on a virtual server, a GRID application client and any Client having the choice in several connection points for data or resource exchange.

3. Uses Cases For Using Multiple Costs

The ALTO protocol specification in [RFC7285] focuses on the basic use case of optimizing routing costs in NSP networks. Upcoming use cases however will require both new Cost Types and new Endpoint Properties. Recent ALTO use cases now extend to CDNs, Data centers and other applications that need additional information to select their Endpoints or handle their PIDs. The needed Cost Types depend on the QoE requirements that are specific to the applications. Moreover, the cost values that they may use may change more rapidly than assumed up to now.

The goal of this section is to describe forward looking use case scenarios that are likely to benefit from ALTO, in order to motivate the introduction of new Cost Types and Endpoint Properties as well as the ALTO Multi-Cost extension.

3.1. Use Cases For Using Additional Costs

ALTO Cost Types and Endpoint Properties are registered in two registries maintained by IANA. The ALTO Cost Type registry ensures that the Cost Types that are represented by an ALTO Cost Map are unique identifiers, and it further contains references to the semantics of the Cost Type. The ALTO specification registers ’routingcost’ as a generic measure for routing traffic from a source to a destination. In a similar way the ALTO Endpoint Property

Registry ensures uniqueness of ALTO Endpoint Property identifiers and provides references to particular semantics of the allocated Endpoint Properties. Currently the ’pid’ identifier is registered, which serves as an identifier that allows aggregation of network endpoints into network regions. Both registries accept new entries after Expert Review. New entries should conform to the respective syntactical requirements, and must include information about the new identifier, the intended semantics, and the security considerations. One basic example advocating for multiple Cost Type transactions is an Application Client looking for destination Endpoints or Source/ Destination PID pairs yielding jointly the lowest ’routingcost’ and path delay. We hereby assume that ’routingcost’ values report some monetary cost and that the Application Client chooses to rely on the hopcount to reflect the path delay.

3.1.1. Delay Sensitive Overlay Applications

The ALTO working group has been created to allow P2P applications and NSPs a mutual cooperation, in particular because P2P bulk file-transfer applications have created a huge amount of intra-domain and congestion on low-speed uplink traffic. By aligning overlay topologies according to the ’routingcost’ of the underlying network, both layers are expected to benefit in terms of reduced costs and improved Quality-of-Experience.

Other types of overlay applications might benefit from a different set of path metrics. In particular for real-time sensitive applications, such as gaming, interactive video conferencing or medical services, creating an overlay topology with respect to a minimized delay is preferable. However it is very hard for an NSP to give accurate guidance for this kind of realtime information, instead probing through end-to-end measurements on the application layer has proven to be the superior mechanism. Still, a NSP might give some guidance to the overlay application, for example by providing statistically preferable paths, possibly with respect to the time of day. Also static information like hopcount can serve as an indicator for the delay that can be expected. Thus a Cost Type that can indicate latency, without the need for end-to-end measurements between endpoints, is likely to be useful.

3.1.2. Selection Of Physical Servers Involved In Virtualized Applications

Virtualized applications in large Datacenters are supported by virtualized servers that actually gather resources distributed on several physical servers. The federation of these resources is often orchestrated by a centralized entity that needs to select the physical servers from or to which it will take resources. This entity can be co-located with an ALTO Client that will request and get the ALTO information on the network formed by the physical servers. The physical servers can be assimilated to endpoints with which the orchestration entity trades application resources or content. These resources include computation resources, storage capacity and path bandwidth between the physical servers.

Here too, the applications that are ran are diverse and may have different and specific QoE requirements. The Endpoint selection typically needs to consider both the computational resources at the Endpoints and the resources e.g. in bandwidth on the transmission paths to or among Endpoints. Thus the application QoE requirements drive the Endpoint selection with more or less weight on QoE specific metrics such as hopcount/delay, bandwidth and other resources, that are typically combined with the routing cost and need to jointly integrate the Endpoint and transmission path perspective in the decision process, which is difficult to do with one single Cost Type.

3.1.3. CDN Surrogate Selection

Another use case is motivated through draft [draft-jenkins-alto-cdn-use-cases-01]. The request router in today’s CDNs makes a decision about the surrogate or cache node to which a content request should be forwarded. Typically this decision is based on locality aspects, i.e. the request router tries to select the surrogate node losest to the client. By using the ’routingcost’ Cost Type, an ALTO server allows an NSP to guide the CDN in selecting the best cache node. This is particularly important as CDNs place cache nodes deeper into the network (i.e., closer to the end user), which requires finer grained information. Finally the provisioning of abstracted network topology information across administrative boundaries gains importance for cache federations.

While distance today is the predominant metric used for routing decisions, other metrics might allow sophisticated request routing strategies. For example the load a cache node sees in terms of CPU utilization, memory usage or bandwidth utilization might influence routing decisions for load-balancing reasons. There exist numerous ways of gathering and feeding this kind of information into the request routing mechanism.

For example, information reporting on the occupation level of a cache could be based on a cost reflecting: its remaining computation resources, its remaining storage capacity w.r.t its capacity in storage or computation resources.

As ALTO is likely to become a standardized interface to provide network topology information, the ALTO server could also provide other information that a request router needs. In the next iterations of this draft we will analyse which of these metrics is suitable as a Cost Type or Endpoint Property for CDN Surrogate Selection, and propose to register them in the respective registries.

3.1.4. Some Proposed Additional Properties And Costs

In addition to CDN caches, Endpoint Properties and Costs can be useful to report an Endpoint's load, given that an Endpoint can as well be a physical server in a datacenter or any entity as defined in Section 2 of this draft.

Proposed new Endpoint properties and costs include:

- an Endpoint Property called "EP-Capacity", reflecting the nominal capacity of this endpoint. This capacity could be split into:
    - EP-Nominal-Memory: the storage capacity of the Endpoint.
    - EP-Nominal-Bandwidth: the capacity of the computation resources of the Endpoint.
- an Endpoint Cost called "EP-Occupied-Capacity", reflecting the currently available resources w.r.t. their nominal capacity. As with EP-Capacity, this can be split into:
    - EP-Occupied-Memory: the remaining storage capacity,
    - EP-Occupied-Bandwidth: the remaining computation resources.

Likewise, new Cost Types are needed to describe the resources of the network paths needed for content transport, in particular the utilized network path bandwidth.

- A Cost Type named 'pathoccupationcost' (POC) can be used to reflect the NP view of the utilized path bandwidth. Such an ALTO Cost Type is likely to have values that change frequently. By no means, as stated in the ALTO requirements, are ALTO Cost types expected to reflect real-time values, as these can be gathered by other mechanisms. Instead, a Cost Type such as 'pathoccupationcost' should be used as an abstraction that may be represented by a statistical value, or be updated regularly at a frequency lower than 'real-time', or be provided according to different time periods or other parameters. A provision mode for time dependent cost values is proposed in [draft-randriamasy-alto-cost-schedule-01]

3.2. Use Cases For Multi-Cost ALTO Transactions

Different Cost Types are suitable for different applications. For example, delay sensitive applications look for both low routing cost and low delay, where as other applications, such as non real time content download, look for moderate delay and minimal losses. On the other hand, applications or entities managing application input information may want, for various reasons to update their ALTO information on several Cost Types. So an ALTO Client may want to mix Cost Types in either ’numerical’ and ’ordinal’ mode, for Cost Types values that can be represented by numerical values.

The Multi-Cost ALTO Services propose to:

- include several Cost Types (and/or Cost Modes) in an ALTO client’s Cost Map and Endpoint Cost request,

- provide several Cost Type values (and/or Cost Mode) in an ALTO server’s response, instead of one.

The primary reasons to use Multi-Cost ALTO are:

- Optimizing time and bandwidth: a single ALTO response with a Multi-Cost cost map with three separate Cost Type values takes much less network bandwidth, and fewer CPU cycles, than three separate ALTO requests for three complete single-cost cost maps. The motivation also holds for the Endpoint Cost Service. Multi-Cost ALTO services can straightforwardly provide a more complete set of cost information.

- Facing unpredictable and/or rapid value changes: an ALTO client can get a consistent snapshot of several different rapidly-varying Cost Type values.

3.2.1. Optimized Endpoint Cost Service

The Endpoint Cost Service (ECS) provides cost information about both the application Endpoint resources and the networking resources used to access those Endpoints. In addition, the ECS may be invoked in "short term" situations, that is for frequent requests and/or requests requiring fast responses. For the ECS, the server’s response is restricted to the requested Endpoints, and so is much smaller than a complete Cost Map. Therefore the ECS can be invoked for ’nearly-instant’ information requests, and is particularly well suited for multi-cost ALTO transactions, supporting requests and responses on several Cost Type values simultaneously.

3.2.2. Optimized Filtered Cost Map Service

The set of ALTO Cost Types is not restricted to 'routingcost': ALTO Servers may provide a broader set of metrics. One thing to consider is that the frequency of updates can vary from a Cost Type to another one. Additionally the volume of an entire cost map with values of all available Cost Types, may get rapidly prohibitive for frequent downloads. Given these considerations the Application Client may take better advantage when:

- requesting multi-cost maps filtered w.r.t. Cost Types of compatible update frequencies or dates, which is the responsibility of the Application Client,
- requesting multi-cost maps filtered w.r.t. a restricted set of PID pairs.

In such a case, as with the Endpoint Cost Service, the purpose of a Multi-Cost transaction is to gain time with whatever future use of the received ALTO information. In this case, the Client may mix Cost Types in either 'numerical' and 'ordinal' mode, for Cost Type values that can be represented by numerical values.

3.2.3. Cases Of Unpredicable Endpoint Cost Value Changes

Querying all Endpoint cost values simultaneously is always more time and resources efficient than doing it sequentially.

It becomes a necessity in case of unpredictable and/or rapid value changes on at least one of the ALTO Cost Types. The term 'rapid' here means "Typical update intervals [that] may be several orders of magnitude longer than the typical network-layer packet round-trip time (RTT)", as described in [RFC6708], up to a couple of minutes.

This section provides two examples of a delay sensitive application using 'routingcost' and 'hopcount' to select an Endpoint. The application can choose between two candidate Endpoints, EP1 and EP2. The initial choice at T=1 is EP1. It is assumed that at T=2 events in the network occur that impact both 'routingcost' and 'hopcount'.

These examples illustrate the need to query 'hopcount' and 'routingcost' values at the same time in order to re-evaluate the EP costs w.r.t. the QoE needs of the application. It is assumed that the application triggers regular ALTO requests to get the latest cost values for a list of candidate Endpoints.

In some cases the Application client wants to use the ALTO information to perform multi-variate optimization on several Cost

Type values. In order for the optimization to be reliable, it is recommended that the Cost Type values are provided in ’numerical’ Cost Mode. Therefore the requested Cost Mode for the applicable Cost Types SHOULD be ’numerical’.

3.2.3.1. Case Of A Multi-Cost ALTO Query Upon A Route Change

In Figure 1, initially at time T=1, the application has chosen EP1 rather than EP2, despite the higher routing cost, because EP1 has a "better" (lower) ’hopcount’ value and despite the higher routing cost and possibly because the application has set a higher weight to ’hopcount’.

At a time T=2, the route to EP1 changes. The ALTO Server information is accordingly updated. The ALTO client makes its next request to update the cost values for ’routingcost’ and ’hopcount’ on EP1 and EP2. It appears that EP1 has now a hopcount value of 3, the same than for EP2 while its routing cost is higher.

The application realizes that there is no more benefit in keeping interacting with EP1 and therefore switches to EP2, that now has the same hopcount but a lower routing cost.

```
   T = 1 : EP1: routingcost = 40, hopcount = 2
            EP2: routingcost = 30, hopcount = 3

            EP1 is selected because application is time-sensitive and
            metric ’hopcount’ has a higher weight

                                              .-----.
                 O ---------- O ------------- | EP2 |
               /                              ‘-----’
             /
           /                               .-----.
     Source ----------------------- O ---- | EP1 |
                                           ‘-----’


   T = 2 : EP1: routingcost = 40, hopcount = 3
           EP2: routingcost = 30, hopcount = 3

           - Route to EP1 has changed. Hopcount is now 3

           ==> EP2 is selected because routingcost is lower than for
           EP1, with the same hopcount value
                                             .-----.
                 O ---------- O -------------| EP2 |
               /  \                          ‘-----’
             /     ‘-----.
           /              ‘------.         .-----.
   Source ---------- X --------- [O] ---- | EP1 |
                                           ‘-----’
```

Figure 1: Endpoint re-selection using Multi-Cost ALTO request on updated cost values, upon a chnage in the route.

3.2.3.2. Case Of A Multi-Cost ALTO Query Upon A Cost Value Change

```
  T = 1 : EP1: routingcost = 30, hopcount = 2
           EP2: routingcost = 30, hopcount = 3
           ==> EP1 is selected because application is time-sensitive and
               hopcount metrics has higher weight

                                             .-----.
                O ---------- O ------------- | EP2 |
              /                              '-----'
            /
          /                               .-----.
        O ------------------------ O ---- | EP1 |
                                          '-----'


  T = 2 : EP1: routingcost = 40, hopcount = 2
          EP2: routingcost = 30, hopcount = 3
          Routingcost to EP1 has increased. Hopcount is the same.
          ==> Delay sensitive applications willing to minimize hopcount
              remain with EP1 while other applications may remain
              with EP2, that now has a lower routingcost.

                                            .-----.
                O ---------- O -------------| EP2 |
              /                             '-----'
            /
          /                               .-----.
        O ------------------------ O ---- | EP1 |
                                          '-----'
```

Figure 2: Endpoint selection using 2 Cost Types with joint request on updated cost values and for delay sensitive applications.

4. ALTO Protocol Updates Needed To Support Multi-Cost Transactions

To allow running Multi-Cost ALTO Services some minor changes in the base protocol are needed. The main updates consist of changing the JSON type of the value taken by the costs and add a few members to the objects describing the information resources, client requests and server responses to Multi-Cost information services.

As written in the introduction, this section relies on Section {11.2.3.6} of the ALTO protocol draft, see [RFC7285] , which allows protocol extensions to encode cost values as the 'JSONValue' data type.

4.1. List Of ALTO Protocol Updates Required And Recommended

The following updates to the ALTO protocol ([RFC7285]) are required or recommended to support multi-cost ALTO transactions. The new resulting JSON formats are specified in the next sections. Section references ({##}) are to the ALTO protocol document.

The applicable ALTO information resources are: Cost Map, Filtrered Cost Map and Endpoint Cost Map, becoming respectively Multi-Cost Map, Filtered Multi-Cost Map and Endpoint Multicost Map provided with the same media-type.

- Updates required in the format of objects member(s):

  - Objects DstCosts (to destination PIDs, {11.2.3.6}) and EndpointDstCosts (to destination Endpoints, {11.5.1.6}): the cost value member evolves to an array of JSONValues.

  - Object ReqFilteredCostMap {11.3.2.3} and ReqEndpointCostMap {11.5.1.3}: a new member named "multi-cost-types" is introduced. This member is an array of 1 or more cost types for which a Multi-Cost ALTO Client requests values. Each cost type of the array is encoded as specified in {10.7}.

- Updates recommended in the object structure:

  - The capabilities for the Filtered Cost Map Service {11.3.2.4} and the Endpoint Cost Map Service {11.5.1.4} need to be extended with a new member entitled "max-cost-types" giving the maximum number of Cost Types allowed in a Multi-Cost request and response.

  - The capabilities for the Multi Cost Map Service need to include a new member named "multi-cost-type-names" and giving the list of Cost Types that are provided in a Multi-Cost Map requested via a GET method.

  - The capabilities for the Cost Map Service, the Filtered Cost Map Service {11.3.2.4} and the Endpoint Cost Map Service {11.5.1.4} need to be extended with a new member named "multi-cost-type-names" and giving the list of Cost Types that may be included in the constraints member of a request.

  - In a Server response to {11.3.2.6} filtered cost map request and {11.5.1.6} and to filtered endpoint cost service request: a new member named "multi-cost-types" and described above is added to the "meta" field of the response.

- Rules required on object member description:

  * Order in which the multiple cost values are provided in the responses.

4.2. Updates Required In The Member Format Of Objects

This section specifies the changes in the object member format that are required to enable multi-cost ALTO transactions.

The term Single Cost qualifies the items as they are specified in the current ALTO protocol.

4.2.1. Cost Value Encoded In array of JSON values

The fundamental change to support multi-cost is to encode the cost values as an array of JSONValues. This way, the cost between two PIDs or two Endpoints can be represented in a generic way:

- with several Cost Types,

- with Cost Types whose value can each be encoded with any type of JSON value.

For example, a multi-cost value represented with Cost Types (assuming they are supported by the ALTO Server):

```
["num-routingcost", "num-hopcount", "string-status"]
```

will be encoded in the following JSON Array in a Multi Cost ALTO response:

```
[23, 6, "medium"]
```

The objects impacted by the encoding of ALTO Multi-Cost values in a JSONArray are: DstCosts and EndpointDstCosts. Full specification will be provided in later sections of this draft.

4.2.2. Scalar 'cost-type' Member Replaced By Array 'multi-cost-types' Member

In the base protocol, the various single-cost-map services use a scalar "cost-type" member in the "meta" section to indicate the cost metric and cost mode of the returned values.

In Multi-Cost ALTO, the multi-cost-map services use an array member named "cost-types" instead. The array elements are in the same format as the "cost-type" member in single cost maps, and the order corresponds to the order of values in the array values in the multi-cost map.

Alternatively, we could use the same member name, but define it as an array for multi-cost services. This would simplify some things for a client, but complicate others. Overall, we believe it is easier for a client to use a new member name than to overload the type of an existing member name.

4.2.3. Rule On Cost Value Order In ALTO Reponses

The cost values each Source/Destination pair MUST be provided in the same order as in the array of Cost Types. This way, the cost type values are provided without any ambiguity on the Cost Type they report on.

4.3. Updates Recommended In The Object Structure

Objects MultiCostMapCapability {11.2.3.4} and FilteredMultiCostMapCapability {11.3.2.4}: are extended with:

- a new member a new member entitled "max-cost-types" giving the maximum number of Cost Types allowed in a Multi-Cost request and response and giving the maximum number of Cost Types in a response. The default value is set to 1 to avoid a multi-cost aware client requesting a multi-cost map from a server that does not support them.

- a new member named "multi-cost-type-names" and giving the list of Cost Types that are provided in a Multi-Cost Map requested via a GET method.

5. Extended Constraints On Multi-Cost Values

This draft proposes to extend the constraint tests in the base protocol to allow tests on the various costs in a request, and to allow more general predicates.

NOTE: Constraint tests on multiple cost metrics are useful even when retrieving single costs, and we expect there will be proposals to add multi-cost constraint tests to the ALTO protocol, relating to the extensions proposed in this draft. Draft [draft-lee-alto-app-net-info-exchange] proposes in particular extensions to query values on a metric M1 with constraints on other metrics {M2, ... Mk}, that adds an interesting feature to extend ALTO constraints. This motivates the need to augment the capabilities in the IRD of the Filteredt Multi-Cost Map and Endpoint Multi-Cost Map with the extensive list of Cost-Types that may be included in the constraints of requests.

The base ALTO protocol allows optional contraints in the input parameters to a request for a Filtered Cost Map or the Endpoint Cost Service. The ’constraints’ member is an array of expressions that all apply to the (single) requested Cost Type. The encoding of ’constraints’ member, is fully specified in Section 11.3.2.3 of the base protocol as follows:

```
 A constraint contains two entities separated by whitespace:
(1) an operator,’gt’ for greater than, ’lt’ for less than,
’ge’ for greater than or equal to, ’le’ for less than or equal to,
or ’eq’ for equal to
(2) a target cost value. The cost value is a number that MUST be
defined in the same units as the Cost Type indicated by the costtype
parameter
    ...
If multiple ’constraint’ parameters are specified, they are
interpreted as being related to each other with a logical AND.
```

Such a specification covers multiple predicates on one metric such as:

’routingcost’ values belong to [6, 20)

5.1. Use Cases For Multi-Cost Multi-Operator Constraints

Suppose that an application uses information on the ALTO Cost Types ’hopcount’ and ’routingcost’. This application may want to select paths or Endpoints with bounds on values for both ’hopcount’ and ’routingcost’. For instance solutions meeting a constraint like:

’hopcount’ values in [6,20) OR ’routingcost’ values in [100,200]

Moreover, this application may be ready to make compromises and to select paths or Endpoints by bounding their cost values according to two options:

1. either solutions with moderate ’hopcount’ and high ’routingcost’, for instance: ’hopcount’ values in [6,20] AND ’routingcost’ values in [100,200],

2. or solutions with higher ’hopcount’ and moderate ’routingcost’, for instance: ’hopcount’ values in [20,50] AND ’routingcost’ values in [30,100].

5.2. Extended constraints in Multi-Cost ALTO

This draft proposes to support the two above mentioned use cases by extending the scope of constraints in two ways:

- allow the ’constraint’ member to be applicable to multiple Cost Types,

- allow the multiple constraints to be related to each other by both logical AND and logical OR.

The two options would be covered by a logical expression like:

```
    [(’hopcount’ ge 6) AND (’hopcount’ lt 20) AND
    (’routingcost’ ge 100) AND (’routingcost’ le 200)]
OR
    [(’hopcount’ ge 20) AND (’hopcount’ le 50) AND
   (’routingcost’ ge 30) AND (’routingcost’ le 100)]
```

A simple encoding of multi-cost constraints for such expressions is specified in Section 5.3.3 of this draft, describing the input parameters to request for Filtered Cost Map. This specification is applicable to the EP Cost service as well.

6. Protocol Extensions for Multi-Cost ALTO Transactions

This section proposes extensions of the ALTO protocol to support Multi Cost ALTO Services or provide additional ALTO information. It integrates discussions on the ALTO mailing list.

If an ALTO client desires information on several Cost Types, then instead of placing as many requests as costs, it may request and receive all the desired Cost Types in one single transaction.

The ALTO server then, provided it supports the requested Cost Types, and provided it supports multi-cost ALTO transactions, sends one single response where for each {source, destination} pair, the cost values are arranged in an array, where each component corresponds to a specified Cost Type. The correspondence between the components and the Cost Types is implicitly indicated in the ALTO response. Indeed, the values in the Cost values MUST be provided in the same order as in the array of cost types indicated in the response.

The following ALTO services have corresponding Multi-Cost extensions:

- Information Resources Directory: extended with multi-cost related URIs and associated capabilities.

- Cost Map Service: extended with the Multi-Cost Map Service,

- Cost Map Filtering Service: extended with the Multi-Cost Map Filtering Service,

- Endpoint Cost Lookup Service: extended with the Endpoint Multi-Cost Lookup Service.

6.1. Information Resources Directory

When the ALTO server supports the provision of information on multiple costs in a single transaction, the Information Resources Directory will list the corresponding resources. The media type remains the same as in the current ALTO protocol.

6.1.1. Example of Multi-Cost specific resources in the IRD

The following is an example Information Resource Directory returned by an ALTO Server and containing Multi-Cost specific services: the Multi-Cost Map Service, Filtered Multi-Cost Map and the Endpoint Multi-Cost Service. It is assumed that the IRD contains usual ALTO Services as described in the example IRD of the current ALTO protocol. In this example, the ALTO Server can additionally provide Multi-Cost Services in a specific folder of "alto.example.com" called "multi". This folder contains the Multi-Cost Maps, Filtered Multi-Cost Maps as well as the Endpoint Multi-Cost Service.

In this example, the ALTO IRD exposes Multi-Cost capabilities on cost types "num-routingcost", "num-hopcount", "num-pathoccupationcost", that can be combined in a request. The values on these metrics are provided in numerical mode. Values provided for cost-type string are in "string" mode.

For the "filtered-multicost-map" resource and the "endpoint-multicost-map" resource, the IRD exposes in its capabilities a member noted "testable-cost-types" that is the list of cost-types that are allowed to be included in the constraints of a request. Note that this set may be different than the set "multi-cost-type-names". The "endpoint-multicost-map" resource provides cost-values for Cost Types "num-routingcost", "num-hopcount" and "str-status" and supports constraints on "num-routingcost", "num-hopcount", "num-pathoccupationcost" where as it does not provide values on "num-pathoccupationcost" and does not supports constraints on "str-status".

```
GET /directory HTTP/1.1
   Host: alto.example.com
   Accept: application/alto-directory+json,application/alto-error+json


   HTTP/1.1 200 OK
   Content-Length: [TODO]
   Content-Type: application/alto-directory+json

   {
     "meta" : {
       "cost-types" : {
          "num-pathoccupationcost" : {
            "cost-mode" : "numerical",
            "cost-metric" : "pathoccupationcost"
          },
          "str-status" : {
            "cost-mode" : "string",
            "cost-metric" : "status"
          },
          "num-routing" : {
            "cost-mode" : "numerical",
            "cost-metric" : "routingcost"
          },
          "num-hopcount" : {
            "cost-mode" : "numerical",
            "cost-metric" : "hopcount"
          },
           .....
           Other ALTO cost types as described
           in current ALTO Protocol
           .....
         },
         "default-alto-network-map" : "my-default-network-map"
       },
     "resources" : {
         "my-default-network-map" : {
           "uri" : "http://alto.example.com/networkmap",
           "media-type" : "application/alto-networkmap+json"
         },
         "numerical-routing-cost-map" : {
            .....
            Single-cost Services as described
            in current ALTO Protocol
            .....
         },
         "rc-hc-multicost-map" : {
           "uri" : "http://alto.example.com/multi/costmap",
           "media-types" : ["application/alto-costmap+json"],
           "uses" : [ "my-default-network-map" ],
           "capabilities" : {
             "multi-cost-type-names" : [ "num-routing", "num-hopcount" ]
           }
         },
         "filtered-multicost-map" : {
           "uri" : "http://alto.example.com/multi/costmap/filtered",
           "media-types" : ["application/alto-costmap+json" ],
           "accepts" : ["application/alto-costmapfilter+json" ],
           "uses" : [ "my-default-network-map" ],
           "capabilities" : {
             "cost-constraints" : true,
             "max-cost-types" : 3,
             "cost-type-names" : [ "num-routingcost",
                                   "num-hopcount",
                                   "num-pathoccupationcost" ],
             "testable-cost-types": ["num-routingcost",
                                     "num-hopcount",
                                     "num-pathoccupationcost" ]
           }
         },
         "endpoint-multicost-map" : {
           "uri" : "http://alto.example.com/multi/endpointcost/lookup",
           "media-types" : [ "application/alto-endpointcost+json" ],
           "accepts" : [ "application/alto-endpointcostparams+json" ],
           "uses" : [ "my-default-network-map" ],
           "capabilities" : {
             "cost-constraints" : true,
             "max-cost-types" : 3,
             "cost-type-names" : [ "num-routingcost",
                                   "num-hopcount",
                                   "str-status" ],
             "multi-cost-type-names" : [ "num-routingcost",
                                         "num-hopcount",
                                         "str-status"],
             "testable-cost-types": ["num-routingcost",
                                     "num-hopcount",
                                     "num-pathoccupationcost" ]
         }
       }
     }
   }
```

6.2. Multi-Cost Map Service

This section introduces a new media-type for the Multi-Cost map. For each source/destination pair of PIDs, it provides the values of the different Cost Types supported for the Multi-Cost map, in the same order as in the list of Cost Types specified in the capabilities.

A Multi-Cost Map MAY be provided by an ALTO Server.

Note that the capabilities specify implicitly the order in which the different Cost Type values will be listed in the Cost Map.

The Cost Type values in the responses are encoded as a JSONArray of cost values for the different Cost Types.

Note that values in a Multi-Cost map are arrays of values of the various Cost Types. If the ALTO server does not have the value for a particular Cost Type for a source/destination PID pair, the server MUST use ’null’ (a reserved JSON symbol) for that location in the array. If the ALTO server does not have a value for any of the Cost Types for a given source/destination pair -- that is, if the array would be a list of nulls -- then the ALTO server MAY omit the array for that source/destination pair.

6.2.1. Media Type

The media type is "application/alto-costmap+json".

6.2.2. HTTP Method

This resource is requested using the HTTP GET method.

6.2.3. Input Parameters

None.

6.2.4. Capabilities

The capabilities of the URI providing this resource are defined by a JSON object of type FilteredCostMapCapabilities::

```
object {
     JSONString multi-cost-type-names<1..*>;
   } MultiCostMapCapabilities;
```

with members

multi-cost-type-names The Cost Type names returned by this map.

An ALTO Server MUST support all of the Cost Types listed here. Note that an ALTO Server may provide multiple Cost Map Information Resources, each with different capabilities.

An ALTO Server supporting the Multi-Cost Map service MUST support the Cost mode ’numerical’ for all supported Cost Types encoded with the ’JSONNumber’ type.

A full cost map resource capabilities has either "cost-type-names" or "multi-cost-type-names", but not both. The former means it returns a Single Cost Map, the latter means it returns a multi-cost Map. Since this resource is requested via the GET method, the Server returns what it returns and the client has no choice.

6.2.5. Uses

The Resource ID of the Network Map which defines the PIDs used in this Multi Cost Map. An ALTO Server MUST NOT define two Multi Cost Maps with the same Network Map and set of Cost Types.

6.2.6. Response

The "meta" field of a Cost Map response MUST include the "dependent-vtags" key, whose value is a single-element array to indicate the Version Tag of the Network Map used, where the Network Map is specified in "uses" of the IRD.

The "meta" MUST also include the member "multi-cost-types", which is a JSONArray of the CostTypes in this Multi Cost Map.

The data component of a Multi Cost Map response is named "cost-map", which is a JSON object of type CostMapData, as defined in {11.2.3.6} of the ALTO protocol. This is identical to the format of the ALTO Cost Map response, except that the JSONValues are arrays rather than numbers. The values in the arrays correspond to the Cost Type listed at the same place in the ’multi-cost-types’ array. This array MUST have the same size as the ’multi-cost-types’ array, and the provided values MUST be in the same order as in the ’multi-cost-types’ array.

The returned Multi Cost Map MUST include the required Path Costs for each pair of Source and Destination PID for which this information is available. If a cost value is not defined, the ALTO Server MUST replace that value in the array with the reserved JSON symbol ’null’. If no costs are defined for a pair of Source and Destination PIDs, so the Path Cost would be an array of nulls, the ALTO Server MAY omit the array for that pair.

6.2.7. Example

This example illustrates a ’static’ multi-cost ALTO transaction, where the utilized Cost Types all have ’static’ values. We assume here that the Cost Types available at the ALTO Server are "routingcost" and "hopcount" and the ’numerical’ mode is available for both of them. The "routingcost" may be based on monetary considerations where as the "hopcount" is used to report on the path delay. We also assume that ALTO server does not know the value of the "routingcost" between PID2 and PID3, and hence uses ’null’ for those costs.

```
GET /multicostmap/num HTTP/1.1
Host: alto.example.com
Accept: application/alto-costmap+json,application/alto-error+json

HTTP/1.1 200 OK
Content-Length: [TODO]
Content-Type: application/alto-costmap+json

{
 "meta" : {
   "dependent-vtags" : [
     {"resource-id": "my-default-network-map",
      "tag": "3ee2cb7e8d63d9fab71b9b34cbf764436315542e"
     }
   ],
   "multi-cost-types" : [
     {"cost-mode": "numerical", "cost-metric": "routingcost"},
     {"cost-mode": "numerical", "cost-metric": "hopcount"}
   ]
 }
 "cost-map" : {
   "PID1": { "PID1":[1,0],    "PID2":[5,23],   "PID3":[10,5] },
   "PID2": { "PID1":[null,5], "PID2":[1,0],    "PID3":[15,9] },
   "PID3": { "PID1":[20,12],  "PID2":[null,1], "PID3":[1,0]  }
 }
}
```

6.3. Filtered Multi-Cost Map

A Multi-Cost Map may be very large. In addition, an Application Client assisted by the ALTO Client does not necessarily need the Cost Types for all the source/destination PID pairs.

Therefore applications may more likely use Cost Map information filtered w.r.t. the Cost types as well as the source/destination pairs of PIDs. This section specifies Filtered Multi-Cost Maps.

A Filtered Multi Cost Map is a Cost Map Information Resource for which an ALTO Client may supply additional parameters limiting the scope of the resulting Cost Map. A Filtered Multi Cost Map MAY be provided by an ALTO Server.

6.3.1. Media Type

The media type is "application/alto-costmap+json".

6.3.2. HTTP Method

This resource is requested using the HTTP POST method.

6.3.3. Input Parameters

Input parameters are supplied in the entity body of the POST request. This document specifies the input parameters with a data format indicated by the media type "application/alto-costmapfilter+json", which is a JSON Object of type ReqFilteredMultiCostMap, where:

```
   object {
     PIDName srcs<0..*>;
     PIDName dsts<0..*>;
   } PIDFilter;

   object {
     CostType    multi-cost-types<1..*>;
     JSONString  constraints<0..*>; [OPTIONAL] - TO BE UPDATED
     JSONArray   or-constraints<0..*>;   [OPTIONAL]
     PIDFilter   pids;                   [OPTIONAL]
   } ReqFilteredMultiCostMap;
```

with members:

multi-cost-type-names The array of requested Cost Types for the returned costs.
Each listed Cost Type MUST be one of the supported Cost Types

indicated in this resource’s capabilities.

constraints

As specified in section {11.3.2.3} of RFC7285.
The Client MUST specify this member in its requests for single cost services as specified in RFC7285.
The Client MUST NOT specify this member in requests for multi-cost services.
The Client MUST NOT specify both a ’constraint’ and an ’or-constraints’ parameter.
NB: THIS TEXT ON SUPPORT OF BASE PROTOCOL SINGLE COST CONSTRAINTS WILL BE UPDATED IN NEXT VERSIONS

or-constraints

Defines an array of arrays of constraint strings.
This parameter MUST NOT be specified if this resource’s capabilities indicate that constraint support is not available.
A constraint string is an array of additional constraints.
That is the constraint strings of the array are related by logical ANDs. Each string in the
constraint array MUST contain three entities separated by whitespace, in the following format:

```
       [index] op value
```

’Index’ is a number between 0 and the number of Testable Cost Types minus 1, and indicates the Cost Type to which this constraint applies. (The square brackets ([]) surrounding ’index’ are required syntactic sugar. They serve as a reminder that ’index’ is an array index, not a value to test, and they avoid unusual-looking constraints such as "1 ge 5".)
’Op’ is an operator: ’gt’ for greater than, ’lt’ for less than, ’ge’ for greater than or equal to, ’le’ for less than or equal to, ’eq’ for equal to, or ’ne’ for not equal to.
’Value’ is a target cost value to compare against the indicated Cost Type. For numeric Cost Types, ’value’ MUST be a number defined in the same units as the Cost Type indicated by ’index’. ALTO servers SHOULD use at least IEEE 754 doubleprecision floating point [IEEE.754.2008] to store the cost value, and SHOULD perform internal computations using double-precision floating-point arithmetic. For string Cost Types, ’value’ MUST be a string enclosed in single quotes (’).
For array-valued Cost Types, ’eq’ is true iff one of the Cost Type values is equal to ’value’, and ’ne’ is true iff none of the Cost Type values are equal to ’value’. The other operators are not defined for array-valued Cost Types.

The "or-constraints" member defines an array of arrays of constraint strings in the format : [index] op value
The ALSO server MUST return costs that satisfy all constraints

in one or more of the inner lists, and no other costs. That is, ’or-constraints’ is the logical OR of ANDs.

pids A list of Source PIDs and a list of Destination PIDs for which Path Costs are to be returned. If a list is empty, the ALTO Server MUST interpret it as the full set of currently-defined PIDs. The ALTO Server MUST interpret entries appearing in a list multiple times as if they appeared only once. If the "pids" member is not present, both lists MUST be interpreted by the ALTO Server as containing the full set of currently-defined PIDs.

6.3.4. Capabilities

The URI providing this resource supports all capabilities documented in Section 6.2.4 (with identical semantics), plus additional capabilities. In particular, the capabilities are defined by a JSON object of type FilteredMultiCostMapCapability:

```
object {
  JSONString   cost-type-names<1..*>;
  JSONString   multi-cost-type-names<1..*>;
  JSONBool   cost-constraints;
  JSONNumber max-cost-types; [OPTIONAL]
} FilteredMultiCostMapCapability;
```

with members:

cost-type-names
The array of cost types available from this service.

multi-cost-type-names
The array of cost types available from this service.
Its resence means that this resource can return
a multi-cost map. A filtered cost map resource can have
either "cost-type-names" or "multi-cost-type-names" or both
in its capabilities. The former means it can return a single
cost map, the latter a multi cost. The Client selects which.

max-cost-types Indicates the maximum number of cost values
the ALTO Server can provide in a multi-cost array of a
Multi-Cost Map.

cost-constraints If true, then the ALTO Server allows cost constraints to be included in requests to the corresponding URI. If not present, this member MUST be interpreted as if it specified false.

Note that a filtered cost map resource can have either "cost-type-names" or "multi-cost-type-names" or both in its capabilities. The former means it can return a single cost map, the latter a multi cost. The Client selects which one its wants.

6.3.5. Uses

The Resource ID of the Network Map which defines the PIDs used in this Filtered Multi Cost Map.

6.3.6. Response

The response is the same format as for the Multi Cost Map Service (Section 6.2.6). The returned Cost Map MUST NOT contain any source/destination pair that was not indicated (implicitly or explicitly) in the input parameters. If the input parameters contain a PID name that is not currently defined by the ALTO Server, the ALTO Server

MUST behave as if the PID did not appear in the input parameters. If any constraints are specified, Source/Destination pairs for which the Path Costs do not meet the constraints MUST NOT be included in the returned Cost Map. If no constraints were specified, then all Path Costs are assumed to meet the constraints.

6.3.7. Example 1

```
   POST multi/multicostmap/filtered HTTP/1.1
   Host: alto.example.com
   Content-Type: application/alto-costmapfilter+json
   Accept: application/alto-costmap+json,application/alto-error+json

   {
     "multi-cost-types" : [
       {"cost-mode": "numerical", "cost-metric": "routingcost"},
       {"cost-mode": "numerical", "cost-metric": "hopcount"}
     ],
     "pids" : {
       "srcs" : [ "PID1" ],
       "dsts" : [ "PID1", "PID2", "PID3" ]
     }
   }


   HTTP/1.1 200 OK
   Content-Length: [TODO]
   Content-Type: application/alto-costmap+json

   {
     "meta" : {
       "dependent-vtags" : [
         {"resource-id": "my-default-network-map",
          "tag": "3ee2cb7e8d63d9fab71b9b34cbf764436315542e"
         }
       ],
       "multi-cost-types" : [
         {"cost-mode": "numerical", "cost-metric": "routingcost"},
         {"cost-mode": "numerical", "cost-metric": "hopcount"}
       ]
     }
     "cost-map" : {
       "PID1": { "PID1": [1,6], "PID2": [5,23], "PID3": [10,5] }
     }
   }
```

6.3.8. Example 2

This is an example of using constraints to restrict returned source/destination PID pairs to those with 'routingcost' between 5 and 10, or 'hopcount' equal to 0.

```
POST multi/multicostmap/filtered HTTP/1.1
Host: alto.example.com
Content-Type: application/alto-costmapfilter+json
Accept: application/alto-costmap+json,application/alto-error+json

{
  "multi-cost-types" : [
    {"cost-mode": "numerical", "cost-metric": "routingcost"},
    {"cost-mode": "numerical", "cost-metric": "hopcount"}
  ],
  "or-constraints" : [ ["[0] ge 5", "[0] le 10"],
                       ["[1] eq 0"] ]
  "pids" : {
    "srcs" : [ "PID1", "PID2" ],
    "dsts" : [ "PID1", "PID2", "PID3" ]
  }
}


HTTP/1.1 200 OK
Content-Length: [TODO]
Content-Type: application/alto-costmap+json

{
  "meta" : {
    "dependent-vtags" : [
      {"resource-id": "my-default-network-map",
       "tag": "3ee2cb7e8d63d9fab71b9b34cbf764436315542e"
      }
    ],
    "multi-cost-types" : [
      {"cost-mode": "numerical", "cost-metric": "routingcost"},
      {"cost-mode": "numerical", "cost-metric": "hopcount"}
    ]
  }
  "cost-map" : {
    "PID1": { "PID2": [5,23], "PID3": [10,5] }
    "PID2": { "PID2": [1,0]                  },
  }
}
```

6.4. Endpoint Multi-Cost Service

The Endpoint Multi-Cost Service provides information on several Cost Types between individual Endpoints.

This service MAY be provided by an ALTO Server. It is important to note that although this resource allows an ALTO Server to reveal costs between individual endpoints, an ALTO Server is not required to do so. A simple alternative would be to compute the cost between two endpoints as the costs between the PIDs corresponding to the endpoints if these values are available for the requested Cost Types.

When the cost values are requested to perform multi-variate numerical optimization and are each available in the ’numerical’ mode, then the ALTO Client SHOULD request the ’numerical’ mode in order to get a reliable result. Note that this consideration is outside the scope of the ALTO protocol as it relates to the responsibility of the ALTO Client and related entries. However common sense lead to warn that a necessary condition for vector ranking method to be reliable is that the components of the processed vectors are numerical and not ordinal values.

6.4.1. Media Type

The media type is "application/alto-endpointcost+json".

6.4.2. HTTP Method

This resource is requested using the HTTP POST method

6.4.3. Input Parameters

Input parameters are supplied in the entity body of the POST request. This document specifies input parameters with a data format indicated by media type "application/alto-endpointmulticostparams+json", which is a JSON Object of type ReqEndpointMultiCostMap:

```
     object {
          TypedEndpointAddr srcs<0..*>; [OPTIONAL]
          TypedEndpointAddr dsts<1..*>;
     } EndpointFilter;

     object{
         CostType    multi-cost-types<1..*>;
         JSONString  constraints<0..*>;      [OPTIONAL] // TO BE UPDATED
         JSONArray   or-constraints<0..*>;   [OPTIONAL]
         EndpointFilter endpoints;
     } ReqEndpointMultiCostMap;
```

with members:

multi-cost-types Defined equivalently to the "cost-types" input parameter of a Filtered Multi Cost Map.

constraints Defined equivalently to the "constraints" input parameter of a Filtered Multi Cost Map.

or-constraints Defined equivalently to the "or-constraints" input parameter of a Filtered Multi Cost Map.

endpoints A list of Source Endpoints and Destination Endpoints for which Path multiple Costs are to be returned. If the list of Source Endpoints is empty (or not included), the ALTO Server MUST interpret it as if it contained the Endpoint Address corresponding to the client IP address from the incoming connection (see Section 10.3 for discussion and considerations regarding this mode). The list of destination Endpoints MUST NOT be empty. The ALTO Server MUST interpret entries appearing multiple times in a list as if they appeared only once.

6.4.4. Capabilities

The capabilities are the same as described in Section 6.3.4.

6.4.5. Uses

As with the ALTO Endpoint Cost Service, the Endpoint Multi Cost Service MUST NOT use a Network Map.

6.4.6. Response

The "meta" field of an Endpoint Multi Cost response MUST include the "multi-cost-types" key, to indicate the Cost Types used.

The data component of an Endpoint Multi Cost response is named "endpoint-cost-map", which is a JSON object of type EndpointCostMapData, as defined in Section 11.5.1.6 of the ALTO protocol. This is identical to the format of the ALTO Cost Map response, except that the JSONValues are arrays rather than numbers. The values in the arrays correspond to the Cost Type listed at the same place in the ’multi-cost-types’ array. This array MUST have the same size as the ’multi-cost-types’ array, and the values in the MUST be in the same order as in the ’multi-cost-types’ array.

6.4.7. Example

This is an example of requesting jointly cost values for "routingcost" and "hopcount" while using constraints to restrict the returned source/destination endpoints to those with ’pathoccupationcost’ between 5 and 10, or ’hopcount’ equal to 0, where ’pathoccupationcost’ and ’hopcount’ respectively have index 2 and 1 in the "testable-cost-types" member of the IRD capabilities of the "endpoint-multicost-map" resource. Only 2 of the 3 requested source/destination pairs meet the constraints.

```
POST multi/endpointmulticost/lookup HTTP/1.1
Host: alto.example.com
Content-Length: [TODO]
Content-Type: application/alto-endpoincostparams+json
Accept: application/alto-endpointcost+json,application/alto-error+json

{
  "multi-cost-types" : [
    {"cost-mode": "numerical", "cost-metric": "routingcost"},
    {"cost-mode": "numerical", "cost-metric": "hopcount"}
  ],
  "or-constraints" : [ ["[2] ge 5", "[2] le 10"],
                       ["[1] eq 0"] ],
  "endpoints" : {
    "srcs": [ "ipv4:192.0.2.2" ],
    "dsts": [
      "ipv4:192.0.2.89",
      "ipv4:198.51.100.34",
      "ipv4:203.0.113.45"
    ]
  }
}
```

```
  HTTP/1.1 200 OK
  Content-Length: [TODO]
  Content-Type: application/alto-endpointcost+json

  {
    "meta" : {
      "dependent-vtags" : [
        {"resource-id": "my-default-network-map",
         "tag": "3ee2cb7e8d63d9fab71b9b34cbf764436315542e"
        }
      ],
      "multi-cost-types" : [
        {"cost-mode": "numerical", "cost-metric": "routingcost"},
        {"cost-mode": "numerical", "cost-metric": "hopcount"}
      ]
    }
    "endpoint-cost-map" : {
      "ipv4:192.0.2.2": {
        "ipv4:192.0.2.89"    : [1, 7],
        "ipv4:203.0.113.45"  : [3, 2]
      }
    }
  }
```

7. IANA Considerations

Information for the ALTO Endpoint property registry maintained by the IANA and related to the new Endpoints supported by the acting ALTO server. These definitions will be formulated according to the syntax defined in Section on "ALTO Endpoint Property Registry" of [RFC7285],

Information for the ALTO Cost Type Registry maintained by the IANA and related to the new Cost Types supported by the acting ALTO server. These definitions will be formulated according to the syntax defined in Section on "ALTO Cost Type Registry" of [RFC7285],

7.1. Information for IANA on proposed Cost Types

When a new ALTO Cost Type is defined, accepted by the ALTO working group and requests for IANA registration MUST include the following information, detailed in Section 11.2: Identifier, Intended Semantics, Security Considerations.

7.2. Information for IANA on proposed Endpoint Properties

Likewise, an ALTO Endpoint Property Registry could serve the same purposes as the ALTO Cost Type registry. Application to IANA registration for Endpoint Properties would follow a similar process.

8. Acknowledgements

The authors would like to thank Vijay Gurbani, Dave Mac Dysan, Dhruv Dhodi and Young Lee for fruitful discussions and comments on this draft and previous versions.

9. References

9.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC5693] "Application Layer Traffic Optimization (ALTO) Problem Statement", October 2009.

9.2. Informative References

[ID-alto-protocol]
""ALTO Protocol" RFC 7285", September 2014.

[RFC6708] "Application-Layer Traffic Optimization (ALTO) Requirements", February 2012.

[RFC7285] R. Alimi, R. Yang, R. Penno (Eds.), , "ALTO Protocol", September 2014.

[draft-jenkins-alto-cdn-use-cases-01]
""Use Cases for ALTO within CDNs" draft-jenkins-alto-cdn-use-cases-01", June 2011.

[draft-lee-alto-app-net-info-exchange]
"Information Exchange for High Bandwidth Applications in TE networks", October 2013.

[draft-randriamasy-alto-cost-schedule-01]
"ALTO Cost Schedule", July 2012.

[draft-randriamasy-alto-multi-cost-05]
"Multi-Cost ALTO", October 2011.

Authors' Addresses

Sabine Randriamasy
Alcatel-Lucent/Bell Labs
Route de Villejust
NOZAY 91460
FRANCE

Email: Sabine.Randriamasy@alcatel-lucent.com

Wendy Roome
Alcatel-Lucent/Bell Labs
600 Mountain Ave, Rm 3B-324
Murray Hill, NJ 07974
USA

Phone: +1-908-582-7974
Email: w.roome@alcatel-lucent.com

Nico Schwan
Thales Deutschland
Lorenzstrasse 10
Stuttgart 70435
Germany

Email: nico.schwan@thalesgroup.com