ALTO Use Case: Resource Orchestration for Large-Scale, Multi-Domain Data Analytics

draft-xiang-alto-multidomain-analytics-00
draft-xiang-alto-exascale-network-optimization-03

Justas Balcas[2], Greg Bernstein[3], Haizhou Du[4], Azher Mughal[5],
Harvey Newman[1], **Qiao Xiang**[6], Y. Richard Yang[6], Jingxuan Zhang[4]

[1] California Institute of Technology, [2] CERN, [3] Grotto Networking,
[4] Tongji University, [5] University of Southern California, [6] Yale University

December 18, 2017, IETF100 ALTO Interim Meeting

Takeaway from IETF99

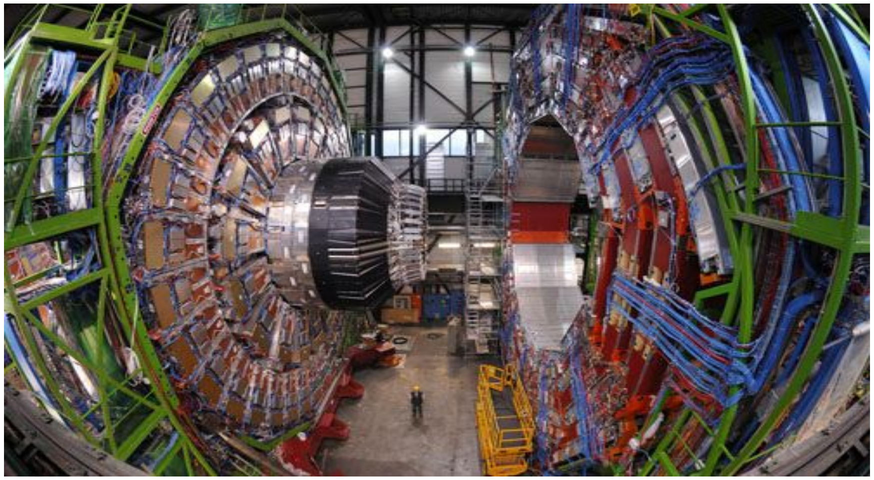
- **Unicorn**: a unified resource orchestration framework for geo-distributed, multi-domain data analytics.
  - Add the detailed workflow for WG review.
  - Add an example to show how ALTO can reveal fine-grained data locality information.
  - Describe how resource view extractor works.

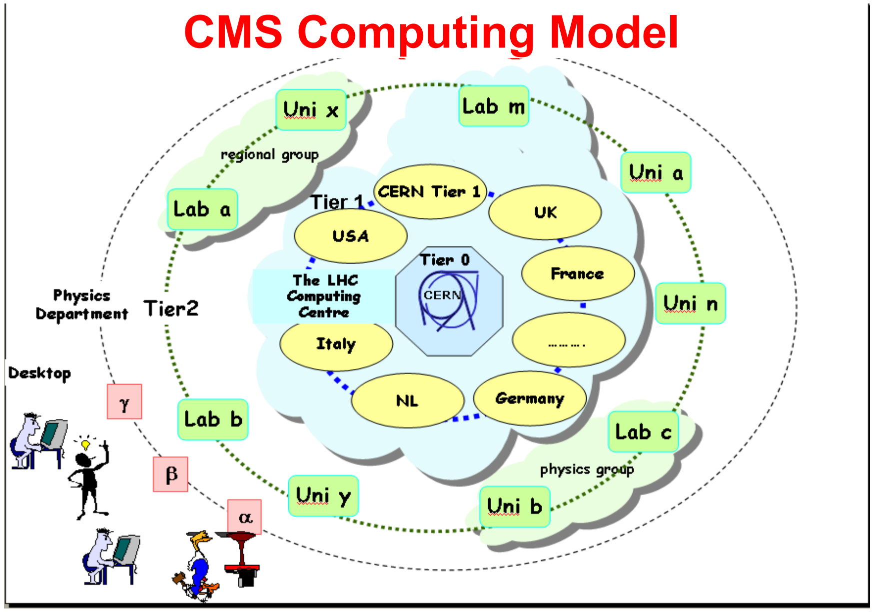
Updates for IETF 100 Interim

- Refactor the document.
  - draft-xiang-alto-multidomain-analytics focuses on the generic system design.
  - draft-xiang-alto-exascale-network-optimization focuses on the implementation and deployment experience of Unicorn.
- System design (draft-xiang-alto-multidomain-analytics).
  - Three-phase resource discovery, i.e., storage/computation resource discovery, path discovery and networking resource discovery.
- System implementation (draft-xiang-alto-exascale-network-optimization).
  - Already demonstrated at SuperComputing 2017 in Nov. 2017.

Multi-Domain, Geo-Distributed Data Analytics

- **Setting**: Different organizations contribute various resources (e.g. , sensing, computation, storage and networking resources) to collaboratively collect, share and analyze extremely large amounts of data.
  - Example: the CMS experiment, coalitions between different organizations, cloud exchange, etc.



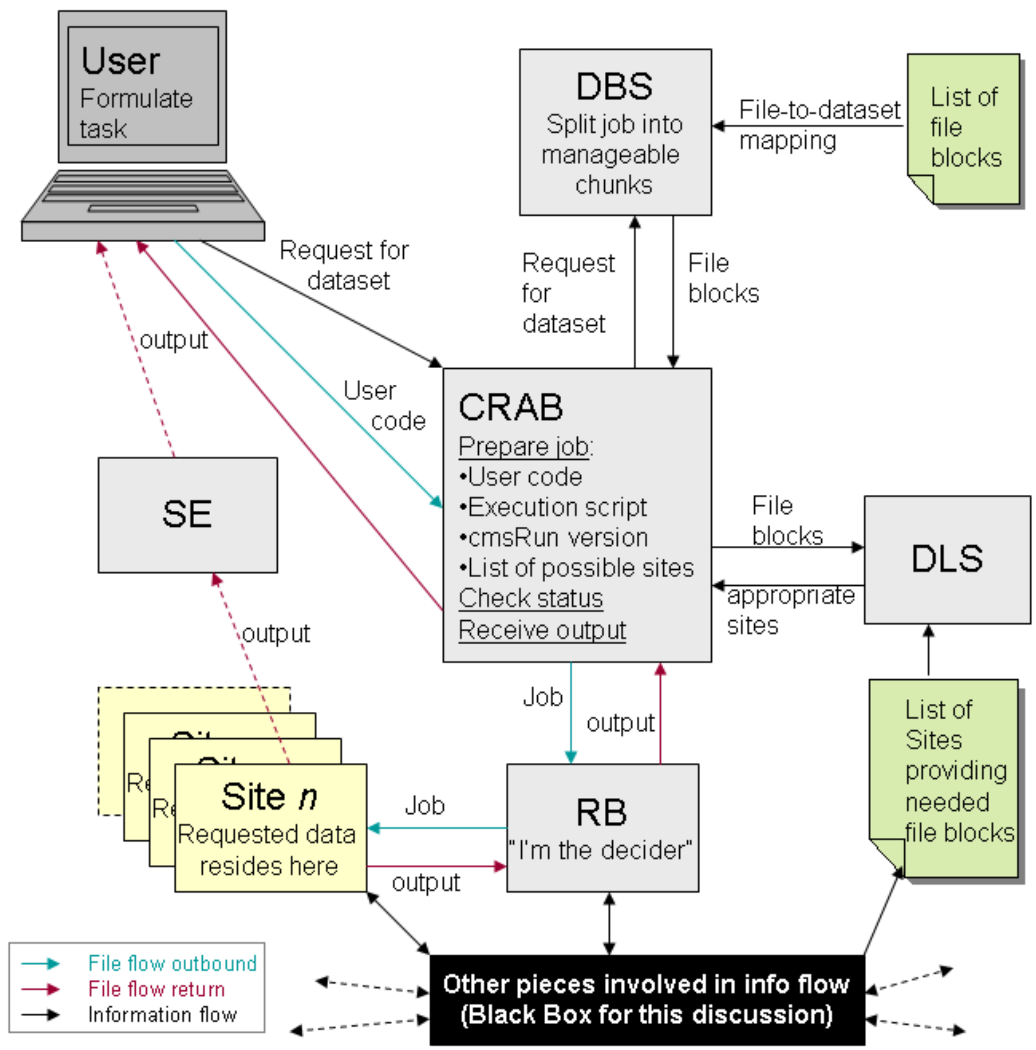


Components

- **Network types**
  - Edge science networks (sites): clusters that provide storage and computation resources for analytics tasks.
  - Transit networks (sites): provide networking resources to move and share large amounts of data among edge science networks.
  - Edge science networks are connected via transit networks.
- **Data analytics tasks**
  - Specified by a 3-tuple (input dataset, program, output site).
  - Decomposable into a set of jobs with a precedence DAG.

Current CMS Data Analytics Work Flow

- **Factors determining data analytics task delay**.
  - Task decomposition (parallelization).
  - Data transmission from input dataset location to computation nodes.
  - Data transmission from computation nodes to output dataset sites.
- Current CMS workflow.
  - Simple, manual parallelization.
  - Opportunistic, network-unaware computation node assignment.
  - Opportunistic, network-unaware output stage out.



Source: https://twiki.cern.ch/twiki/bin/view/CMSPublic/WorkBookAnalysisWorkFlow

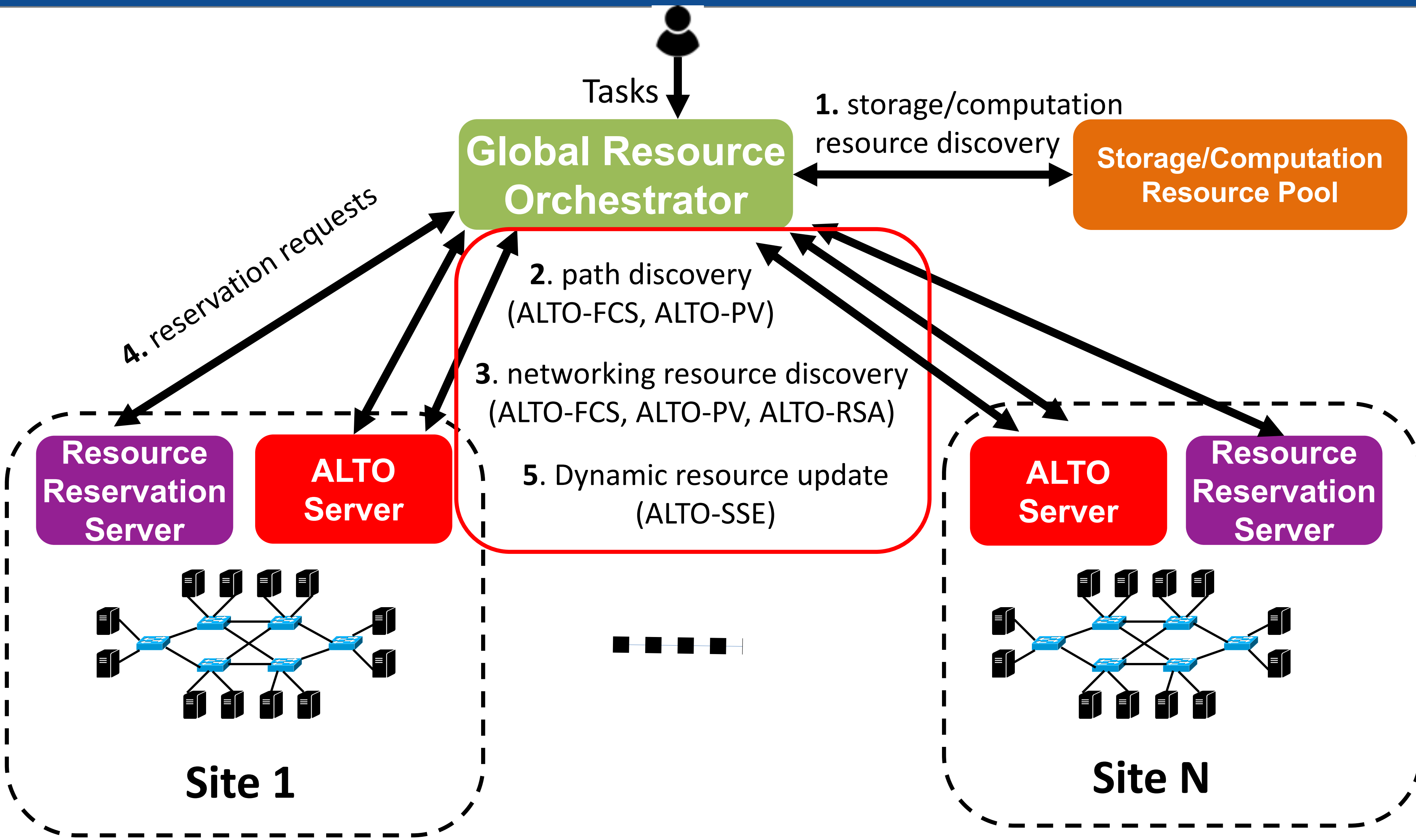
Key Challenge

- Accurate, efficient discovery of resource information, allowing autonomy and protecting privacy of each site.

- **Strawman**: the all-detailed resource graph adopted in single-domain cluster management systems.
  - It cannot be applied due to the exposure of private information and the high overhead.

- **Why ALTO**?
  - ALTO provides on-demand fine-grained information on different resources to support optimal resource orchestration, while preserving privacy of networks.

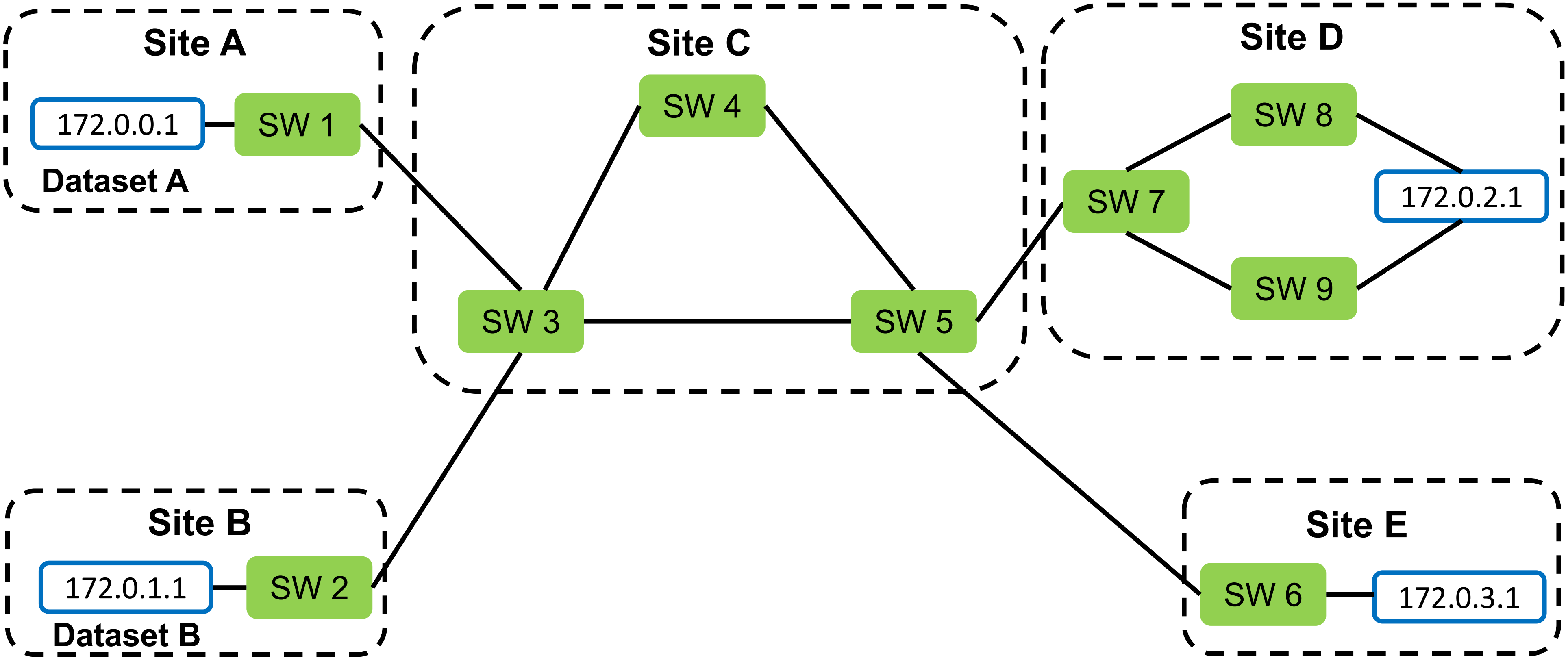
Architecture



Step 1: Storage/Computation Resource Discovery.

- The orchestrator (application) queries DNS servers or a centralized storage/computation resource pool to discover the location (i.e., IP) of candidate storage/computation resources.
- **Design issue**: scalability
  - With a large number of datasets and storage and computation resources, a centralized database may become the single point of failure.
  - **Solution**: use distributed hash table to shard the centralized database into a structured overlay network.

Storage/Computation Resource Discovery: Example



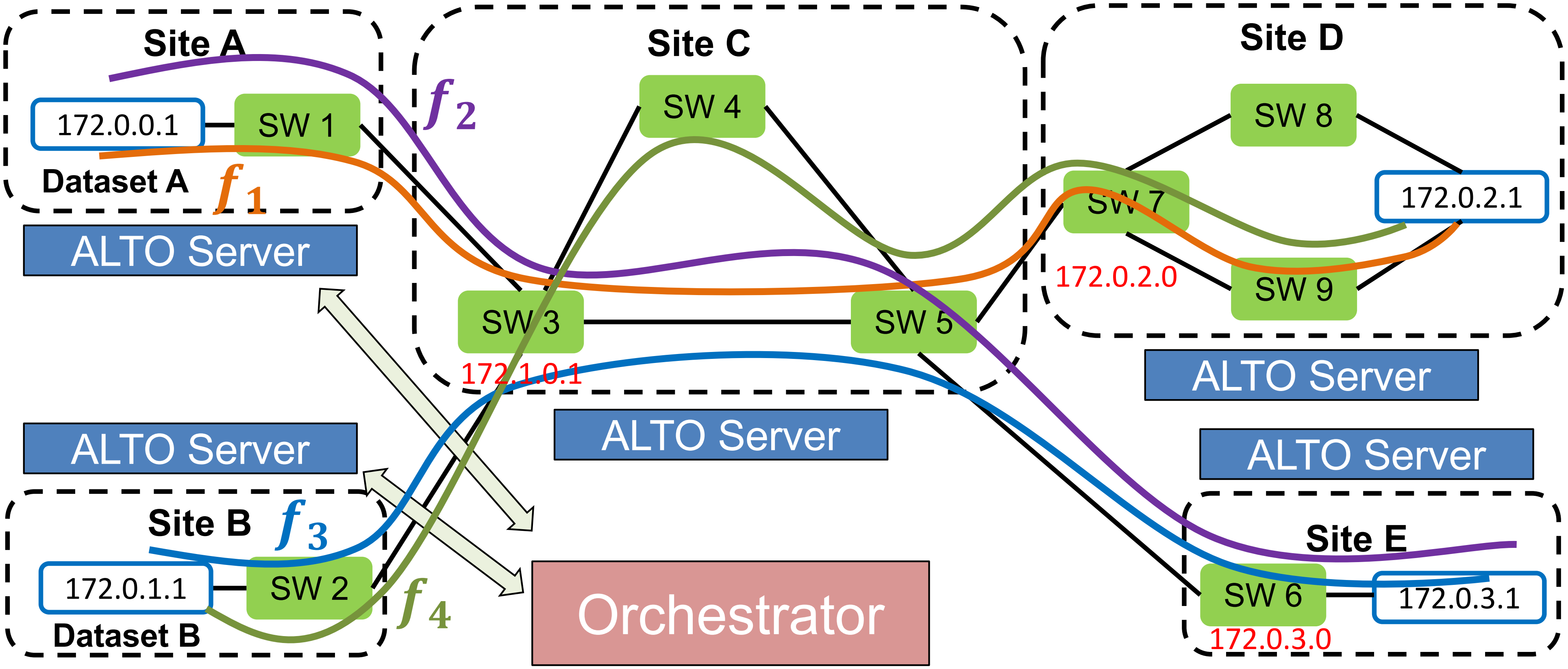
- A user submits a task of two jobs.
  - Job 1 needs dataset A as input and one computation node with no output storage needed.
  - Job 2 needs dataset B as input and one computation node with no output storage needed.
- Storage/computation resource discovery result:
  - {"input-storage": {"job-1": ["172.0.0.1"], "job-2": ["172.0.1.1"]}}
  - {"computation": {"job-1": ["172.0.2.1", "172.0.3.1"], "job-2": ["172.0.2.1", "172.0.3.1"]}}

Step 2: Path Discovery

- The orchestrator sends **flow cost service** (ALTO-Flow Cost Service) queries to ALTO servers to discover the connectivity among input storage resources and computation resources, and among computation resource and output storage resources.
  - Cost type: **path vector** (ALTO-Path Vector).
  - Response: **site-path**, a path vector composes of the ingress IP of each AS along the AS path.
- **Design issue**: Fine-grained flow-based path discovery.
  - BGP provides the basic destination IP-based AS path.
  - SDN provides the potentials for each domain to exchange and make fine-grained flow-based inter-domain routing decisions.
  - **Key challenge**: advertisement explosion due to full instantiation
  - **Solution**: a sub/pub system allowing each domain to only query the routing decisions of a set of flows instead of the whole flow space [1].

[1] Le, *et.al.*, "SFP: Toward a Scalable, Efficient, Stable Protocol for Federation of Software Defined Networks", in DAIS Workshop 2017.

Path Discovery: Example



Query

{ "cost-type":
{ "cost-mode": "array",
"cost-metric": "ane-path" },
"endpoint-flows":
{ "srcs": ["ipv4:172.0.0.1", "ipv4:172.0.1.1"],
"dsts": ["ipv4:172.0.2.1", "ipv4:172.0.3.1"]}
}

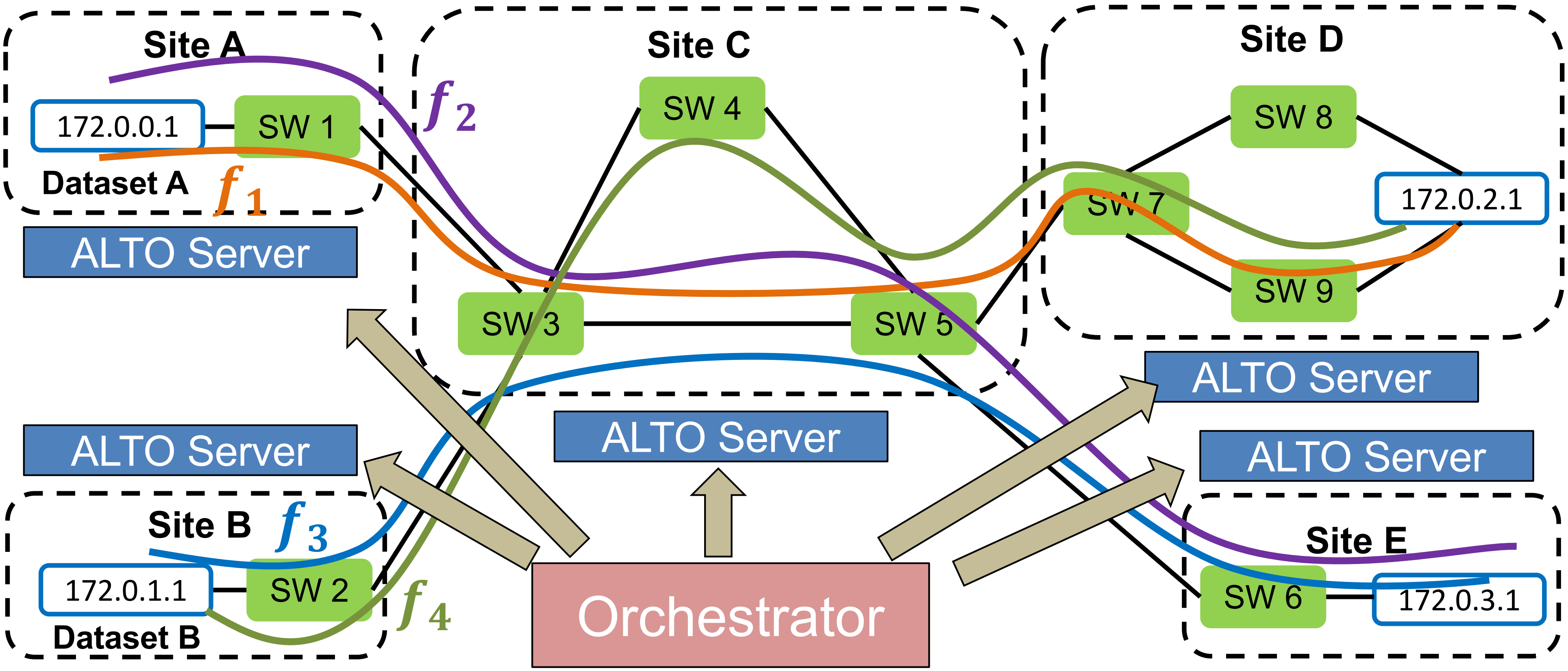
Response

{"endpoint-cost-map":
"ipv4: 172.0.0.1 ": {
"ipv4: 172.0.2.1 ": ["ane:172.1.0.1", "ane:172.0.2.0"],
"ipv4: 172.0.3.1 ": ["ane:172.1.0.1", "ane:172.0.3.0"]},
"ipv4: 172.0.1.1 ": {
"ipv4: 172.0.2.1 ": ["ane:172.1.0.1", "ane:172.0.2.0"],
"ipv4: 172.0.3.1 ": ["ane:172.1.0.1", "ane:172.0.3.0"]}}

Step 3: Networking Resource Discovery

- The orchestrator sends **flow cost service** (ALTO-Flow Cost Service) queries to ALTO servers to discover the shared bottle neck information (i.e., the routing state abstraction) of the set of candidate flows (ALTO-RSA).
  - Cost type: **path vector** (ALTO-Path Vector).
  - Response: **resource state abstraction**, a set of linear inequalities revealing the resource sharing between different (storage, computation) flows.
- Computing minimal, cross-domain resource state abstraction.
  - Strawman: let the orchestrator (application) perform the compression algorithm designed in ALTO-RSA.
    - Unnecessary exposure of network information to the application.
  - **Solution**: a secure multi-party computation protocol that allows ALTO servers from different domains to remove redundant linear inequalities. This solution contains redundant routing state within a limited ALTO servers and only returns the minimal, cross-domain RSA to orchestrator.

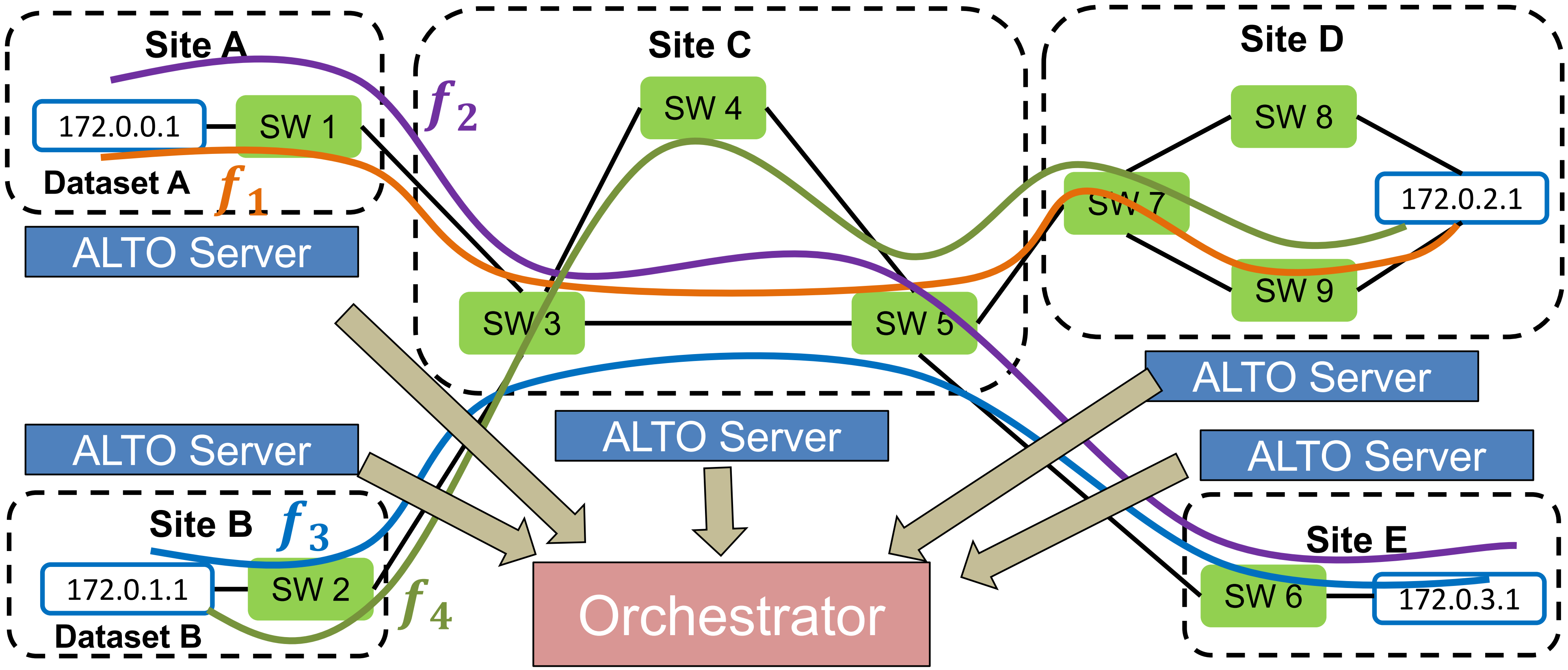
Resource Discovery: Example



Minimal, equivalent Single-Site RSA computed by each ALTO server:

- Site A: $\{f_1 + f_2 \leq 100Gbps\}$
- Site B: $\{f_3 + f_4 \leq 100Gbps\}$
- Site C: $\{f_4 \leq 100Gbps, f_1 + f_2 + f_3 \leq 100Gbps,$ ~~$f_1 + f_2 + f_3 \leq 100Gbps$~~$\}$
- Site D: $\{f_1 + f_4 \leq 100Gbps,$ ~~$f_1 + f_4 \leq 100Gbps\}$~~
- Site E: $\{f_2 + f_3 \leq 100Gbps\}$

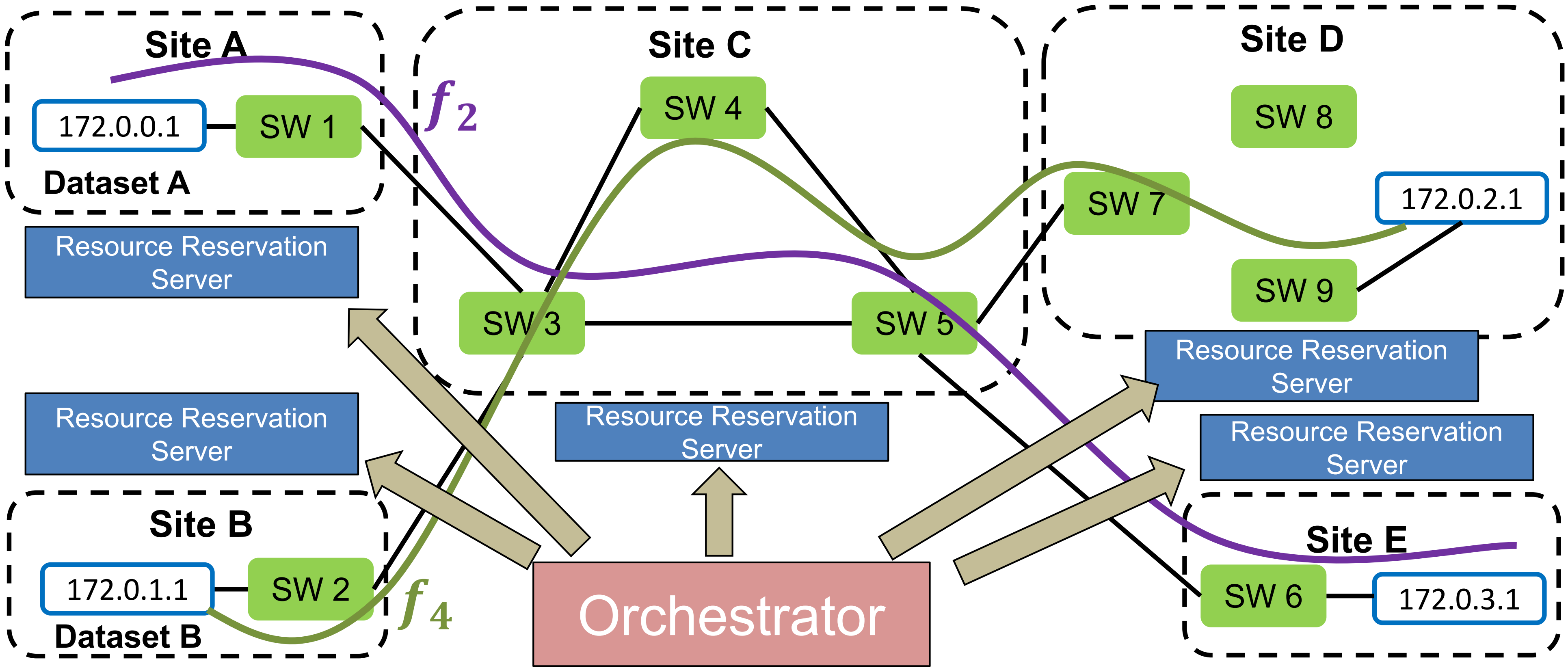
Minimal, Cross-Domain RSA: Example



Minimal, equivalent Across-Site RSA computed via secure computation protocol:

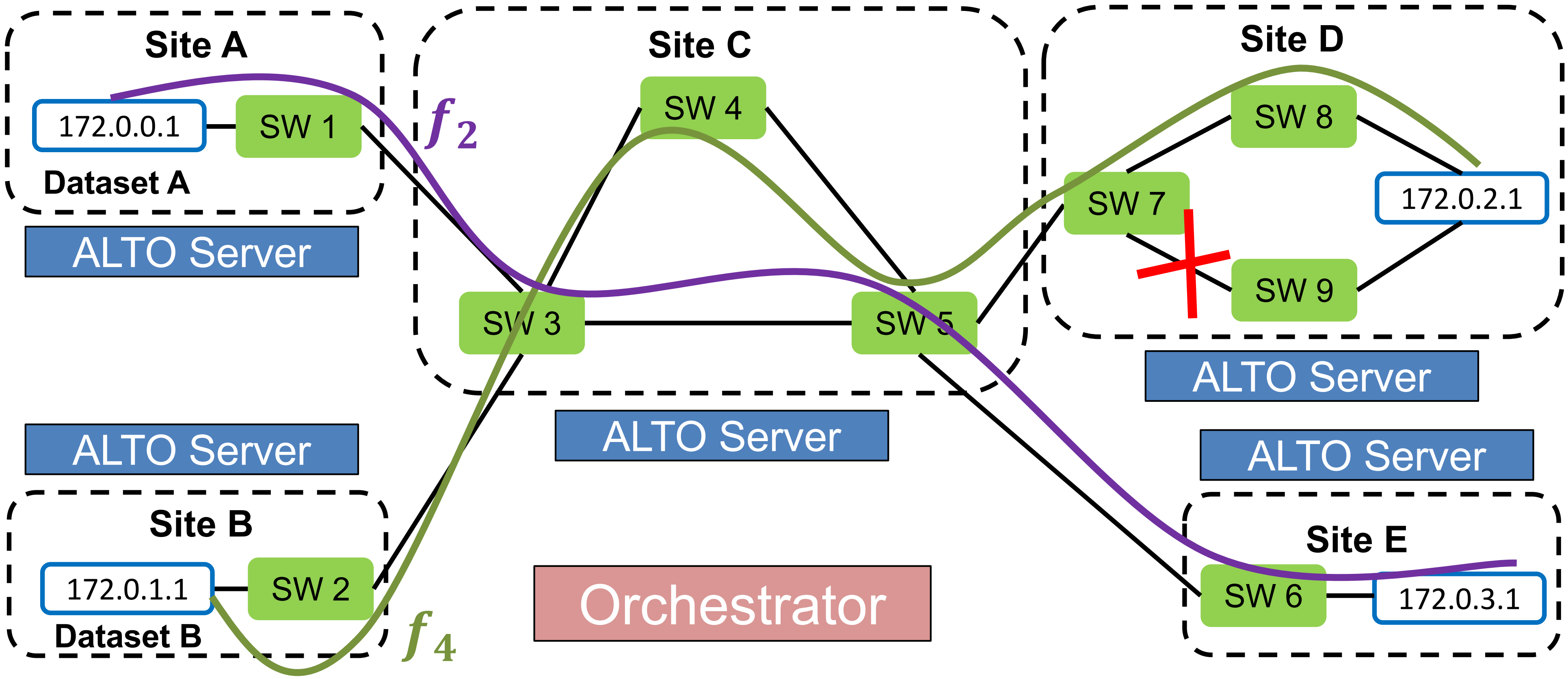
- Site A: ~~$\{f_1 + f_2 \leq 100Gbps\}$~~
- Site B: $\{f_3 + f_4 \leq 100Gbps\}$
- Site C: $\{$~~$f_4 \leq 100Gbps,$~~$f_1 + f_2 + f_3 \leq 100Gbps\}$
- Site D: $\{f_1 + f_4 \leq 100Gbps\}$
- Site E: ~~$\{f_2 + f_3 \leq 100Gbps\}$~~

Step 4: Resource Orchestration and Reservation



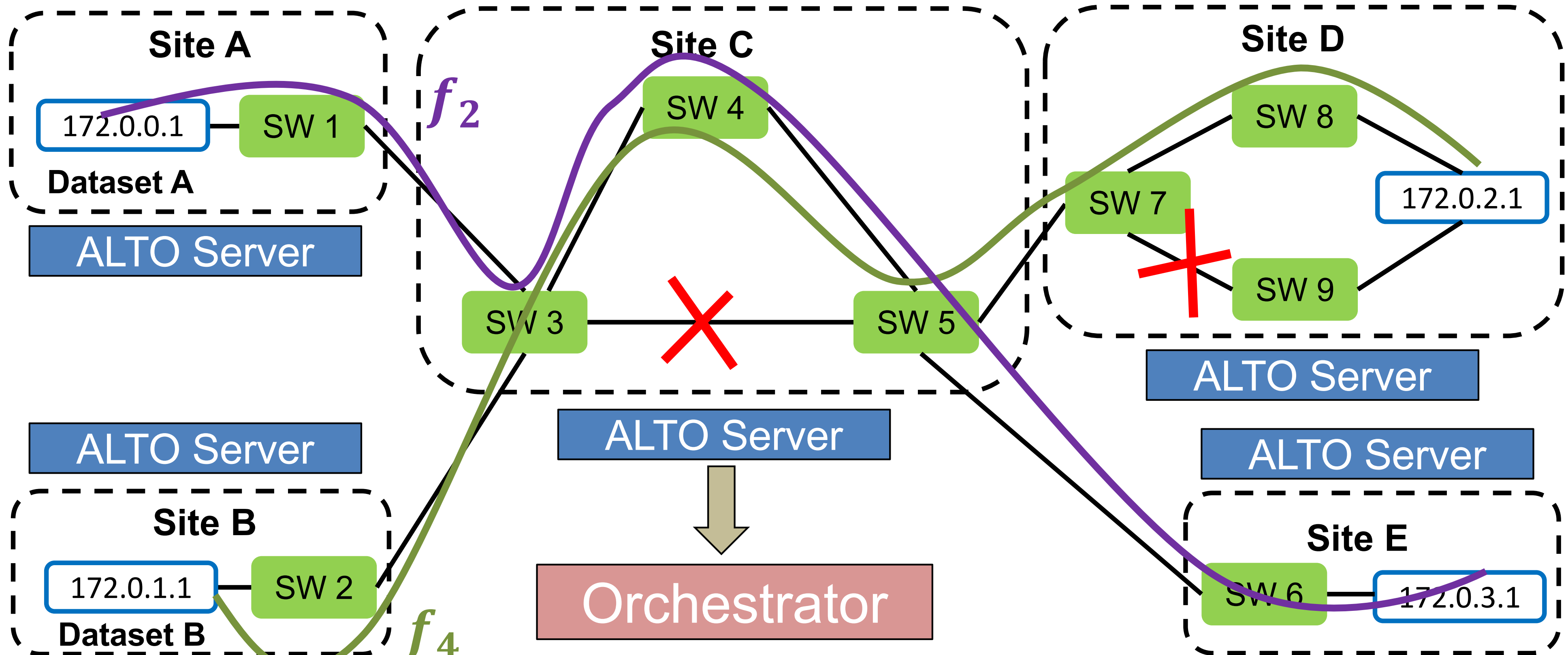
- Using the resource information collected from ALTO servers, the orchestrator computes the optimal orchestration decisions for the submitted jobs, and sends resource reservation requests to each site.
  - *f_2 for Job 1 and f_4 for Job 2, each with a 100Gbps bandwidth.*

Handling Resource Dynamicity: ALTO SSE



- Underlying resource events are transparent to the orchestrator as long as the abstract resource information is not changed.
  - **Event 1**: link $sw7 \rightarrow sw9$ is down.
  - The path of f_4 changes: $\mathrm{sw2} \rightarrow sw3 \rightarrow sw4 \rightarrow sw5 \rightarrow sw7 \rightarrow sw8$.
  - However, the RSA of Site D does NOT change, i.e., $f_4 \leq 100Gbps$.

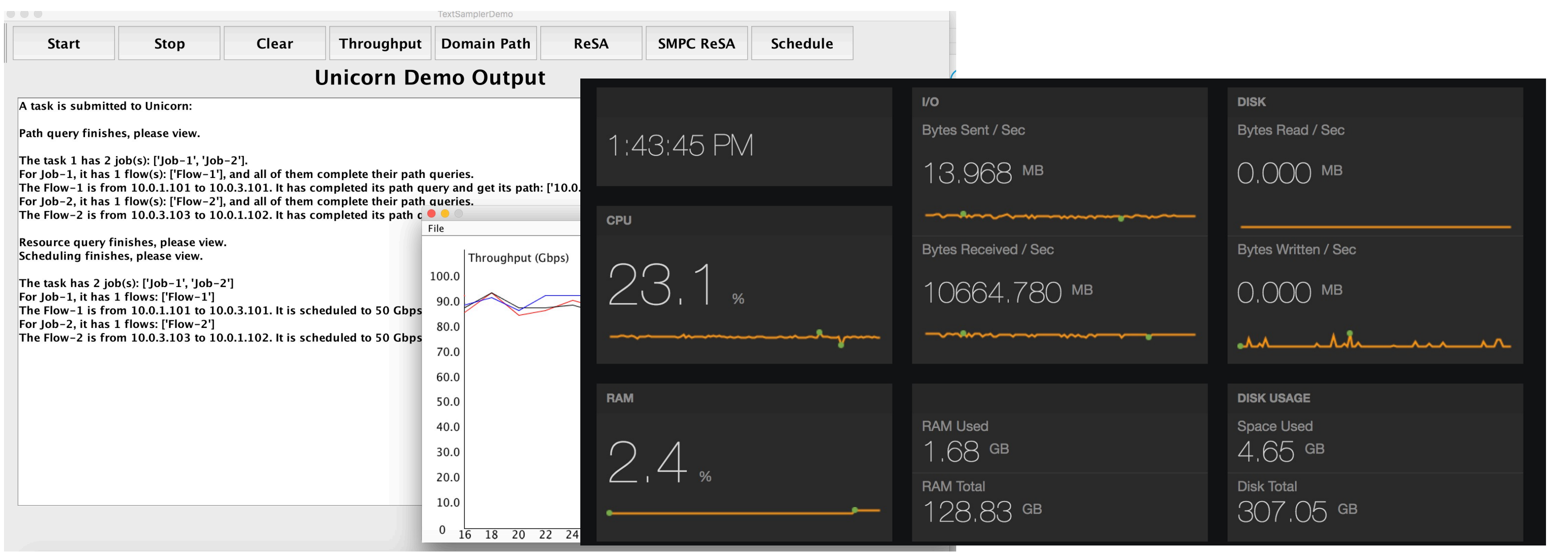
Handling Resource Dynamicity: ALTO-SSE



- When network events changes the abstract resource information, the Unicorn server only sends the updated ReSA, instead of network events, to the orchestrator for updated orchestration decisions
  - **Event 2:** link $sw3$→sw5 is down.
  - The path of f_2 changes: sw2→$sw3$→$sw4$→$sw5$→$sw6$.
  - The RSA of Site C DOES change, i.e., $f_2 + f_4 \leq 100Gbps$.
  - Site C sends the updated RSA to the orchestrator.

Unicorn Implementation and Demonstration

- Orchestrator: ~2700 LoC Python code
- ALTO server: ~3000 LoC Java code
- Resource reservation server:
  - fast data transfer (FDT), FireQoS, OpenvSwitch, etc.
- Network controllers: OpenDaylight, Kytos
  - ONOS and Ryu are under development
- Demonstrated on different topologies at SuperComputing 2017 [2].



[2] Xiang, *et.al.*, "Unicorn: Unified Resource Orchestration for Multi-Domain, Geo-Distributed Data Analytics", in INDIS Workshop 2017.

Importance to ALTO WG

- Unicorn provides a generic design for large-scale, multi-domain data center resource optimization, a major use case of ALTO listed in the WG Charter.
  - In addition to RFC7285, several ALTO extensions (i.e., ALTO-PV, ALTO-RSA, ALTO-FCS, ALTO-SSE) are used in Unicorn to provide resource information for resource orchestration.
- The implementation and deployment experience of Unicorn provides practice guidelines for the use of multiple ALTO services.

Next Steps

- **Draft**
  - Continue to document the design and experience of Unicorn.
  - Add cost calendar services in the design.
  - etc.
- **Milestones**
  - Finish the generic system design by IETF 101.
  - Large scale trial by IETF 102-103.