NSDI'22 Tech Report

Wei Yue

Cloud Lab, Futurewei Technologies

4/28/2022

Some NSDI'22 Facts(not so tech part)

- *April 4-6 2022, Renton, WA*
- *More paper accepted this year(78) than 2021(59), +32%*
- *First time double-track format.*
- *Less in-person attendees than before but comfortable, good to be back from virtual ☺*
- *Lots of interesting papers, will only cover a subset of them here.*

From where?

- UW(8); Princeton(7)
- MIT(4); Duke(4);
- Berkeley(3); UCLA(3)
- Stanford(2); ETH(2); EPFL(2)
- Tsinghua(9);
- PKU(2), USTC(2).

- Microsoft(17)
- FB/Meta(3); Google(3); VMWare(3)
- Alibaba(3), Bytedance(3)
- Futurewei(2); Tencent(2)

Tech sessions

- Cluster Resource Management(3)
- Transport Layer(6);
- Video Streaming(3)
- Programmable Switches(7)
- Security and Privacy(3)
- Network Troubleshooting and Debugging(3)
- Operational Track(7)
- Wireless(7)
- Reliable Distributed Systems(3)
- Raising the Bar for Programmable Hardware(3)
- Testing and Verification(4)
- Sketch-based Telemetry(3)
- Troubleshooting(3)
- Edge IoT Applications(3)
- Cloud Scale Services(4)
- ISPs and CDNs(4)
- Cloud Scale Resource Management(3)
- Data Center Network Infrastructure(3)
- Multitenancy(3)
- Software Switching and Beyond(3)

Some generic takeaways

- Datacenter network and programmable data-plane continues to be challenging areas with lots of active R&D going on, both in academia and industry.

- P4 programming remains to be a painful process for developers, this is natural historically whenever you try to work directly with hardware. Lots of efforts going on to make the process easier, like providing libraries, adding automated processes; using verification tools, etc. ***Or***, argue it has fundamental issues and needs alternative design.

- Sketch-based Telemetry/concurrent sketches is an emergency technique for cloud monitoring.

- Applying other domain knowledge into networking domain to solve newly met issues provide promising results, for examples: applying sketching data structure; using SMT solver or other verification technique to solve real network field issue, quite some papers try to combine cross domain knowledge.

Major Topics to Cover

1. Transport Layer
2. Programmable Data plane
3. Sketch-based Telemetry
4. Other interesting papers

1. Transport Layer Track Summary

Legacy congestion control algorithms are not designed for nowadays traffic patterns in data center and edge applications, which are more latency sensitive, much fatter links and with extremely bursty traffic. Numerous optimizations along these years still can't handle these traffic patterns at optimal level. R&D in this area remains to be active and well needed. Following 6 papers provides interesting solutions for various use cases which could be inspiring for issues we may be facing along the way.

Total 6 papers in this track:

- **PowerTCP** proposes a new congestion control algorithm to specifically handle *dynamic* and *bursty* traffic in data-center network. ***It's a work on top of HPCC which uses INT to collect middlebox feedback for e2e congestion control. PowerTCP then uses collected two dimensional parameters in its algorithm.***
- **RedN** claims to allow RDMA NICs to implement complex offload by exploring existing RDMA verbs(CAS, WAIT and ENBALE) capability to construct a Turing complete set of programming abstractions. ***Interesting work.***
- **FlexTOE** claims to be a flexible, high-performance TOE to SmartNics(Netronome Agilio-CX40, x86 and BlueField) with XDP/eBPF support**.** ***Very impressive work, the challenge remains to be parallelism for stateful in TCP. They claim to use sharding for atomic operation.***
- ***EQDS's*** idea is to move the querying scheme to the sender host coupled with receiver-driven credit scheme. This enables it to support multiple higher(conflicting) layer protocols. ***Worth further deeper dive since it's from Mark Handley & Costin…***
- BFC argue e2e congestion protocol has reached its practical limits. It proposes per-hop per-flow flow control trying to achieve faster reaction, low buffering, high throughput with key ideas: ***Only tract active flows***; ***Dynamic queue assignment*** and ***communicate state across switches***. ***Implemented and tested on Tofino2 P4 programmable switch.***
- **Reframer** is another interesting idea by deliberately delays packeting and reordering them to increase traffic locality. They claim with the cost of us-scale delay of selected packets, **Reframer** increases throughput by up to 84% for some network services.

Extra observation *PowerTCP and EQDS continues to work on better e2e strategies while BFC gives up e2e and try to attack from middle directly*.

Paper list:

1.	*PowerTCP: Pushing the Performance Limits of Datacenter Networks*	*TU Berlin & Univ. of Vienna*
2.	*RDMA is Turning complete, we just didn't know it yet!(RedN)*	*KTH & UW*
3.	*FlexTOE: Flexible TCP Offload with Fine-Grained Parallelism.*	*UW*
4.	*An edge-queued datagram service for all datacenter traffic(EQDS)*	*Mark Handley & Costin Raiciu*
5.	*Backpressure Flow Control(BFC).*	*Tom Anderson UW & MIT*
6.	*Packet Order Matters! Improving Application Performance by Deliberately Delaying Packets.*	*KTH* *Community Award Winners*

PowerTCP - TU berlin and Univ of Vienna

A novel congestion control algorithm, targeting for more fine-grained congestion control by adapting to the bandwidth-window product (henceforth called power).

PowerTCP leverages ***INT*** to react to changes in the network instantaneously without loss of throughput and while keeping queues short.

Due to its fast reaction time, it is particularly well-suited for dynamic network environments and bursty traffic patterns:

best for short flows FCTs;
benefit medium sized flows;
no penalty for long flows;
outperforms under bursty traffic.

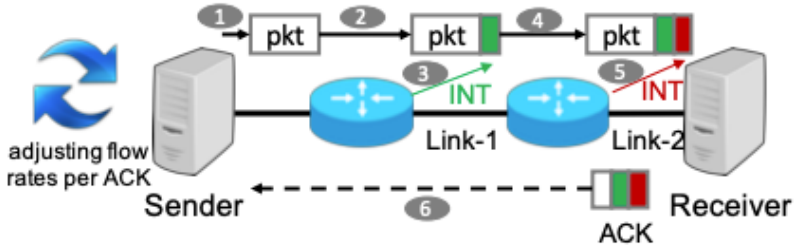


Figure 4: The overview of HPCC framework.

INT leverages here is the same as HPCC, a sender-driven CC framework:
"each packet a sender sends will be acknowledged by the receiver. During the propagation of the packet from the sender to the receiver, each switch along the path leverages the INT feature of its switching ASIC to insert some meta-data that reports the current load of the packet's egress port, including timestamp (ts), queue length (qLen), transmitted bytes (txBytes), and the link bandwidth capacity "(1).

When the receiver gets the packet, it copies all the meta-data recorded by the switches to the ACK message it sends back to the sender. The sender decides how to adjust its flow rate each time it receives an ACK with network load information"(1)

Quantity	Analogy
Total transmission rate (network flow)	Current (λ)
BDP + buffered bytes (network effort)	Voltage (ν)
Current × Voltage	Power (Γ)

Table 1: Analogy between metrics in networks and in electrical circuits. Note that the network here is the ''pipe'' seen by a flow and not the whole network.

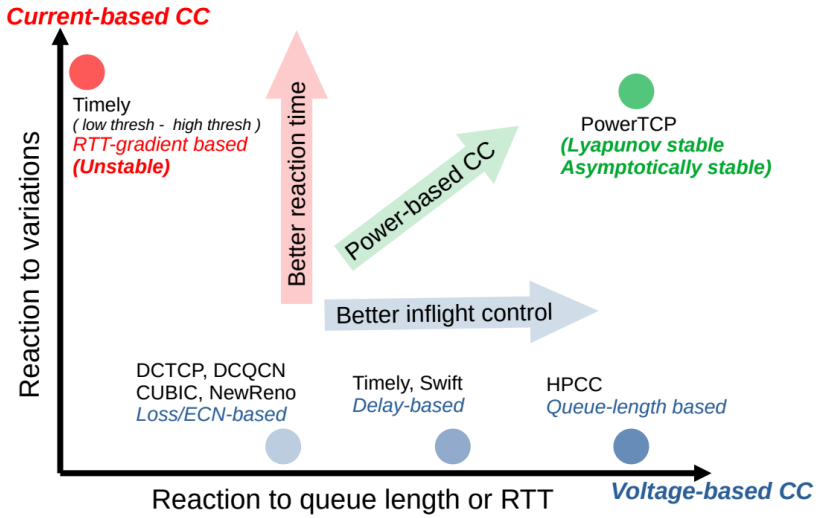


Figure 1: Existing congestion control algorithms are fundamentally limited to a single dimension in their window (or rate) update decisions and are unable to distinguish between two scenarios across multiple dimensions.

1. https://liyuliang001.github.io/publications/hpcc.pdf

FlexTOE: Flexible TCP Offload with Fine-Grained Parallelism

FlexTOE is a flexible, high-performance TCP offload engine (TOE) to SmartNICs, implemented on Netronome Agilio-CX40, also on x86 and Mellanox BlueField.

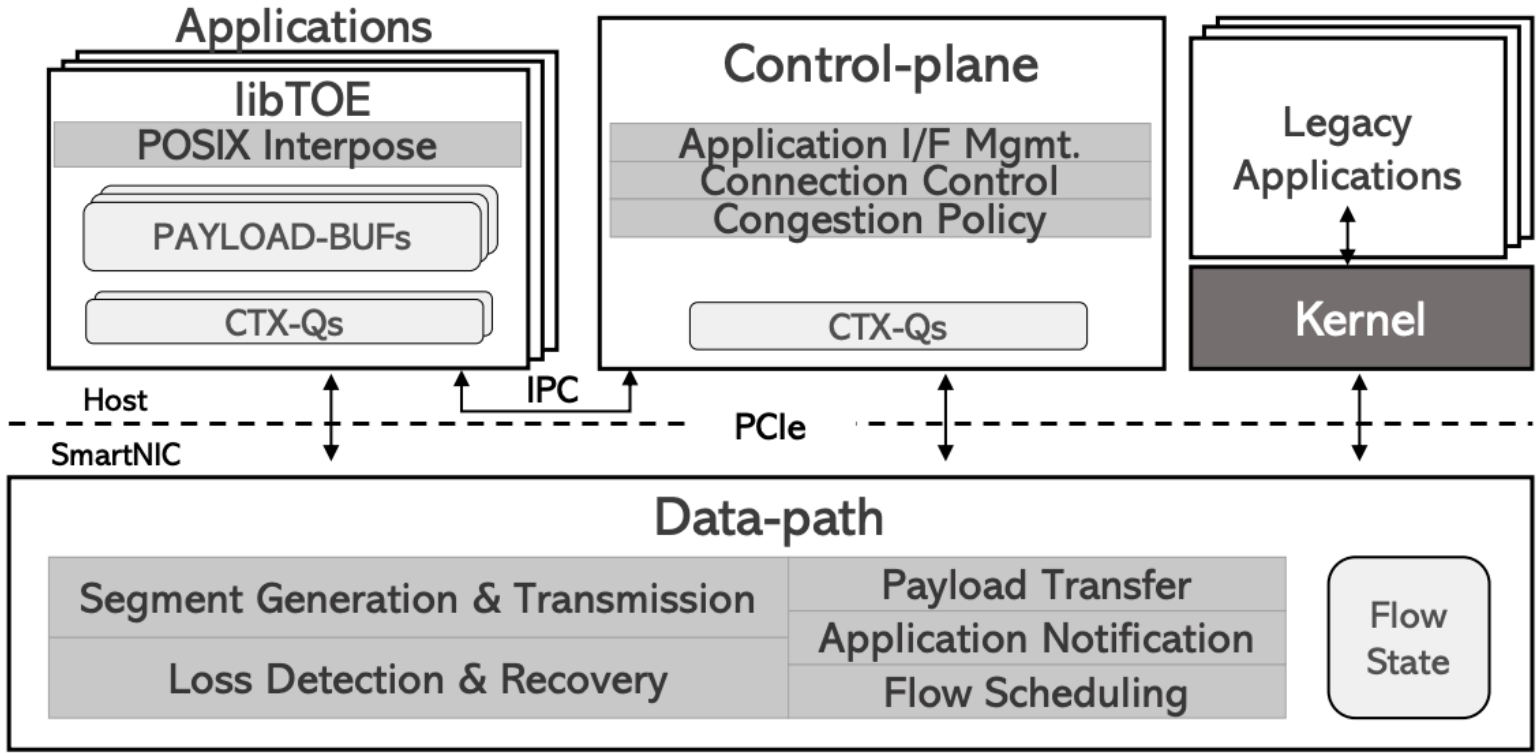


Figure 2. FlexTOE offload architecture (host control-plane).

https://tcp-acceleration-service.github.io/FlexTOE

RedN: RDMA offloading can actually be Turing complete

Claims that normally RDMA requires CPU intervention for complex offloads that go beyond simple remote memory access. As such, the offload potential is limited and RDMA based systems usually have to work around such limitations.

Key contribution in RedN:
Using self-modifying RDMA chains to lift the existing RDMA verbs interface to a Turing complete set of programming abstractions. Take advantage of existing verbs:
CAS: Compare-and-swap (CAS) verb enables dynamic modifying the RNIC execution path by editing subsequent verbs in an RDMA program, using the CAS operands as a predicate
WAIT: allows to halt execution of new verbs until past verbs have completed, providing strict ordering among RDMA verbs
ENABLE: By controlling verb prefetching, ENABLE enforces consistency for verbs modified by preceding verbs. ENABLE also allows us to create loops by re-triggering earlier, already-executed verbs in an RDMA work queue—allowing the NIC to operate autonomously without CPU intervention.

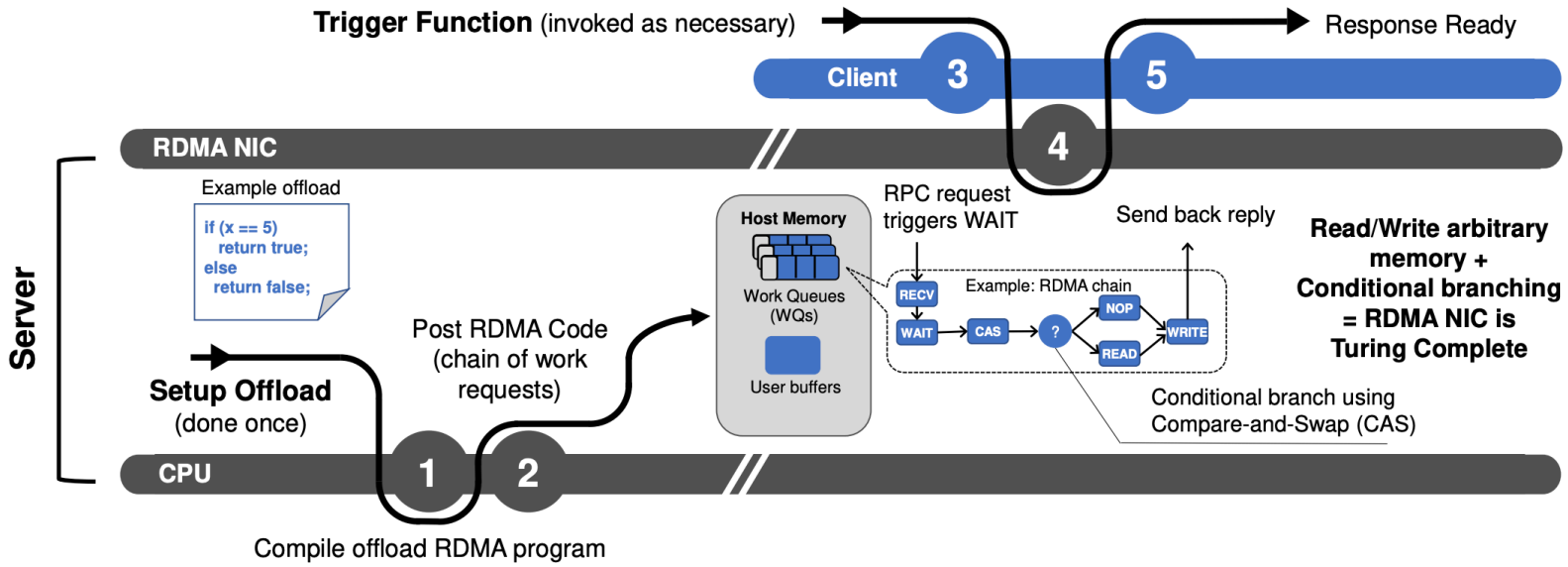


Figure 1: RDMA NICs can implement complex offloads if we allow conditional branches to be expressed. Conditional branching can be implemented by using CAS verbs to modify subsequent verbs in the chain, without any hardware modification.

2. Programmable Switches track

The papers in programmable switch track either try to extend programmable switch(mostly PISA/P4 based) capability from different angles:

- Resource sharing(P4VRM);
- Easier P4 programming, tools for solving resource constraints(P4ALL, Cetus);
- Runtime capability(IPSA/rP4, FlexCore);
- Floating point support to boost performance for applications like SwitchML or Sparks(FPISA).

Or, develop service/applications which takes advantage of programable switches:

- State sharing in data plane(SwiSh);
- Fast and scalable in-network scanner(Imap).

P4 has deep learning curve and exposes more limitations as shown in papers. We have to embrace/study it for programable data plane as P4 has formed a strong eco system with entry barrier removed for network design. The backups are from Operators like ATT, Comcast, China Unicom; Companies like FB, Google, Alibaba and Switch/Chip vendors like Intel(Barefoot), AMD(Xilinx, Pensando), Nvdia(Mellanox), etc.

IPSA/rP4 proposes an alternative to PISA/P4 which emphasizes on runtime capability. This is a continuing effort originated from POF(*proposed at about the same time as P4 published*). While FlexCore tries to work within P4 ecosystem for runtime programmability support, IPSA/rP4 is targeting more complete runtime programmability capability compared with PISA/P4.

Paper list

1. NetVRM: Virtual Register Memory for Programmable Networks(P4VRM)
2. SwiSh: Distributed Shared State Abstractions for Programmable Switches
3. Enabling In-situ Programmability in Network Data Plane: From Architecture to Language
4. Runtime Programmable Switches
5. Imap: Fast and Scalable In-Network Scanning with Programable Switches
6. Unlocking the Power of Inline Floating-Point Operations on Programmable Switches
7. Modular Switch Programming Under Resource Constraints(P4ALL) ***Princeton Jennifer Rexford and David Walker***
8. Cetus: Releasing P4 Programmers from the Chore of Trial and Error Compiling *Tsinghua & Alibaba*

Programable switches

SwiSh: Distributed Shared State Abstractions for Programmable Switches

What?

A distributed shared state management layer for data-plane P4 programs. It enables running scalable stateful distributed network function entirely in the data-plane.

How?

Introduces **SDW**(Strong Delayed-Writes) protocol, which offers consistent snapshots of shared data-plane objects with semantics known as r-relaxed strong linearizability, to implement distributed concurrent sketches with precise error bounds.

Key ideas:

1. *minimizing the buffer space* at the expense of higher bandwidth;
2. *using in-switch packet generator* for implementing reliable packet delivery and synchronization in the data-plane.

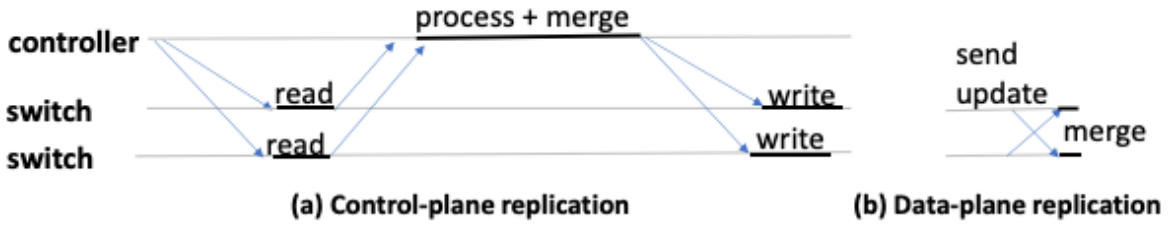


Figure 1: Data-plane vs. control-plane replication

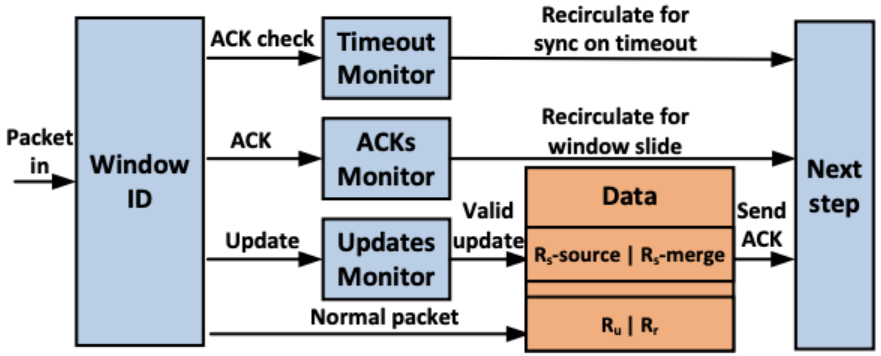


Figure 3: SDW high-level design. Blue boxes are reusable P4 control blocks, while the orange box is application-dependent.

Programable switches

Modular Switch Programming Under Resource Constraints(P4ALL)

P4 enables new kinds of in-network computing that can accelerate distributed applications(NetChain, NetCache) and perform advanced monitoring and telemetry with various data structure, this also brings resource contention due to resource constraints.

Because of this, P4 programming is not easy. It is cumbersome with lots of back-and-forth hard code process .

P4ALL compiler tries to make P4 programming simpler. It enhances p4 by defining elastic data structures that stretch automatically to make optimal use of available switch resources, using ***symbolic primitives*** (that parameterize the size and shape of the structure) and ***objective functions*** (that quantify the value gained or lost as that shape changes).

Data Structure	Used in
Key-value store/ hash table	Precision [6], Sonata [17], Network-Wide HH [19], Carpe [20], Sketchvisor [23], LinearRoad [25], NetChain [26], NetCache [27], FlowRadar [30], HashPipe [41], Elastic Sketch [46]
Hash-based matrix (Sketch)	AROMA [4], Sketchvisor [23], Sketchlearn [24], NetCache [27], Nitrosketch [31], UnivMon [32], Sharma et al. [38], Fair Queueing [39], Elastic Sketch [46]
Bloom filter	NetCache [27], FlowRadar [30], SilkRoad [34], Sharma et al. [38]
Multi-value table	BeauCoup [10], Blink [22]
Sliding window sketch	PINT [5], Conquest [11]
Ring buffer	NetLock [47], Netseer [48]

Figure 1: PISA data structures

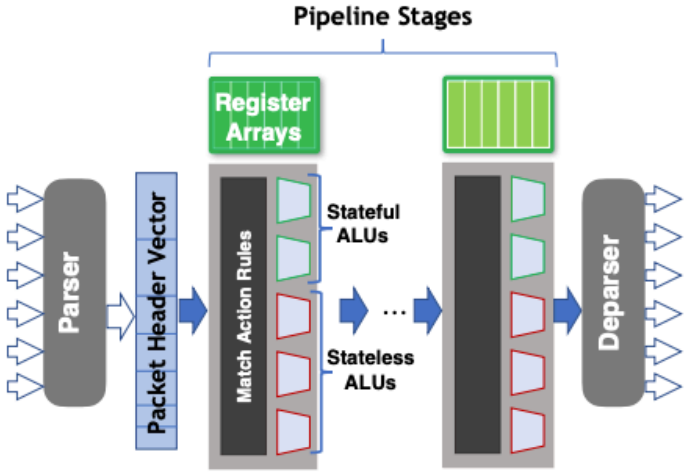


Figure 2: Protocol Independent Switch Architecture (PISA)

Programable switches

Cetus: Releasing P4 Programmers from the Chore of Trial and Error Compiling

Tsinghua & Alibaba

Key contribution

A synthesis system that *translate uncompilable P4 program P into compliable, functionally identical P4 Program P'*

Key Techniques

1. Change the original "long, narrow" DAG to a "short, fat" DAG by removing dependencies on the "longest path" of DAG:

1.1 Dependency Graph building;

1.2 Table merging by dependency removal.

2. Constraint-Based Filter & Optimizer

Using SMT solver to solve constraint issue, to address SMT solver's searching capability limitation issue:

2.1 PHV(Packet Header Vector) encoding approach to greatly reduce the size of SMT formulas;

2.2 two-step solving algorithm to decouples the solving process into table-related resource and variable-related resource solving to speed up the solving process.

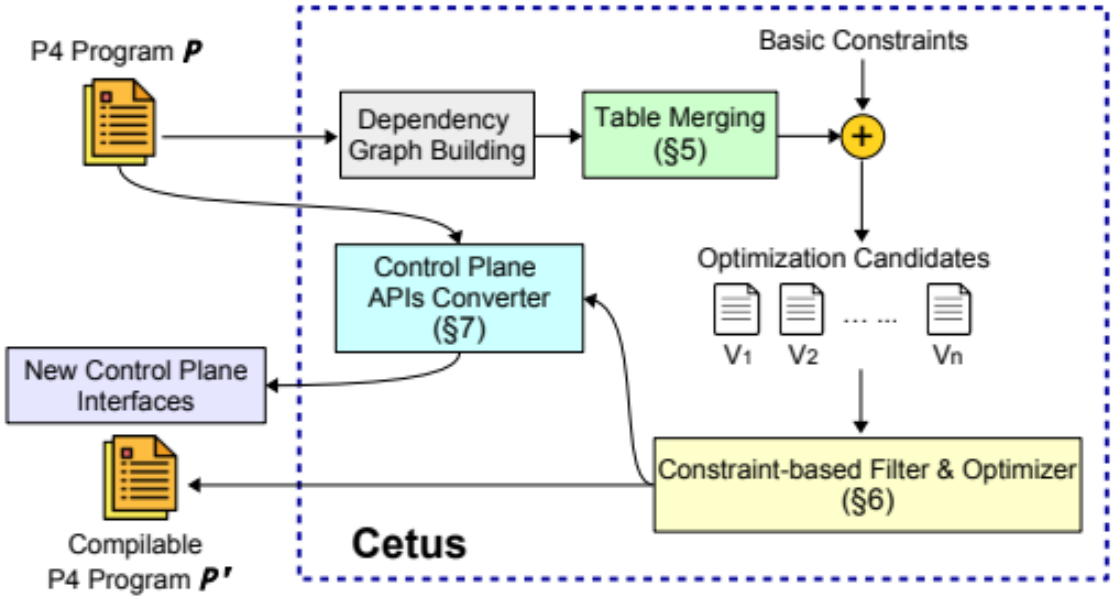


Figure 6: Cetus's workflow overview.

Notes:

Cetus claims P4ALL and some other approaches are not applicable in AliCloud. Alibaba has been applying verification techniques in their various subsystems in recent years.

Programable switches

IPSA/rP4 argue PISA/P4 is incapable of runtime programmability due to its inflexibility in various places. They propose a new IPSA design with 4 major architecture changes to address these limitations.
A P4 extension rP4 is designed for programming IPSA-based devices.

Notes:

It's interesting to compare FlexCore with IPSA in runtime programmability, deeper investigation is needed in this area for us.

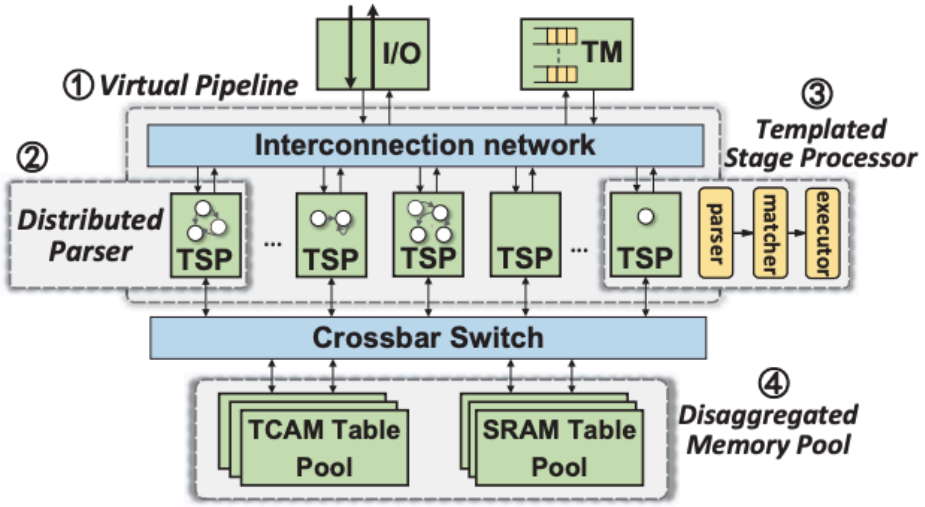


Figure 1: Overview of IPSA.

IPSA vs. PISA (how we solve limitations of PISA)

- Modifying front parser blocks all operations
  Distributed, on-demand, self-contained **parser**
- Binary executable cannot be altered when running
  Template-based stage processor: *parser-matcher-executor* backbone
- Inserting or deleting functions blocks mean processing logic migration
  Virtual pipeline with crossbar (crossbar, CLOS, Bens, BB, and etc)
- Cannot do table refactoring
  Disaggregated memory pool: create, recycle dynamically

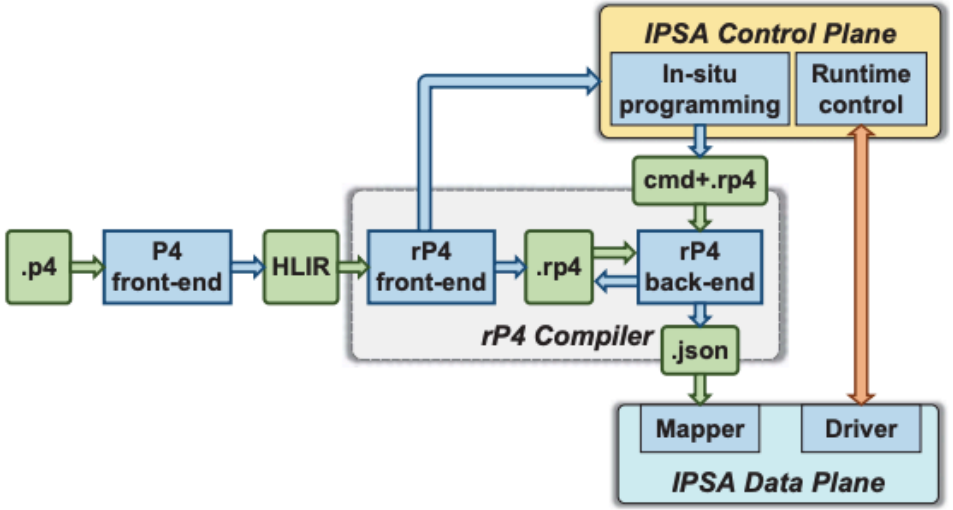


Figure 4: The complete rP4 design flow.

3. Sketch-based Telemetry

First NSDI tech session on the topic, also papers related to sketches in other sessions.

Takeaways

*Sketches have emerged as a promising alternative to traditional sampling-based network telemetry solutions mainly for two reasons: high resource efficiency; accuracy guarantees. This is very attractive for programmable data-plane(**limited hardware resources, high bandwidth flow**) with challenges remain to be solved.*

1. DynATOS

   other approach which offers similar tradeoffs compared to sketches.

   DynATOS adjusts target accuracy(σ) while sketches adjust # counters.

2. SketchLib

   a library for developers to implement sketch algorithms in switch hardware efficiently with resource optimization. Claims to reduces hardware resource footprint up to 96%.

3. HeteroSketch

   A network monitoring framework which claims to enable optimized deployments for >40k nodes with prompt responses to various network telemetries.

   Key approaches: **Automated Profiler, Fast Optimizer**

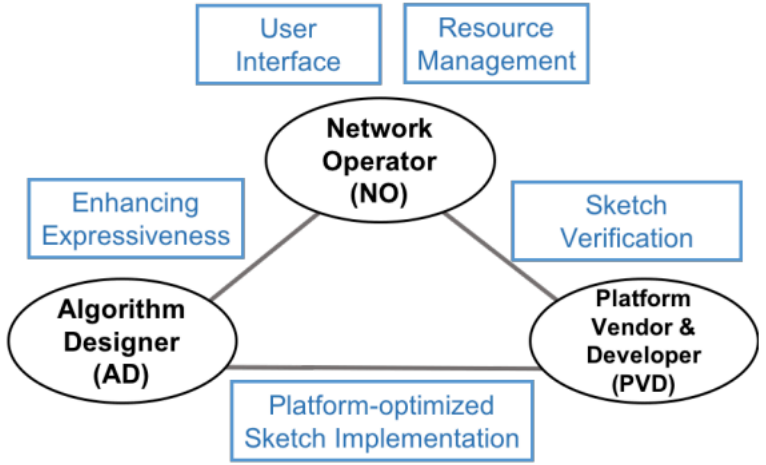


Figure 1: Overview of the problems from the stakeholders in sketch-based telemetry.

Sketch-based Telemetry

Dynamic Scheduling of Approximate Telemetry Queries (DynATOS)

Design Challenges

- D1: Need to approximate generic query operations;
- D2: Need dynamic accuracy estimation without assumptions about traffic; and
- D3: Need to schedule finite hardware resources among dynamic query workload.

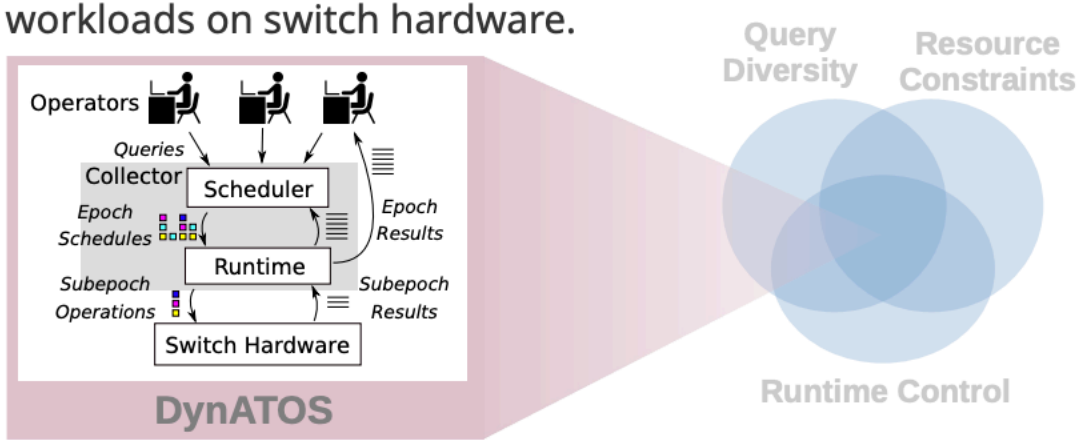
Key Insights

- (1) Time-division approximation:
  - Enables tradeoffs to reduce resource requirements (D1);
  - Can estimate error based on observations alone (D2).
- (2) Scheduling formulation:
  - Realizes tradeoffs enabled by time-division approximation (D3);
  - Efficiently decides how to execute query operations.

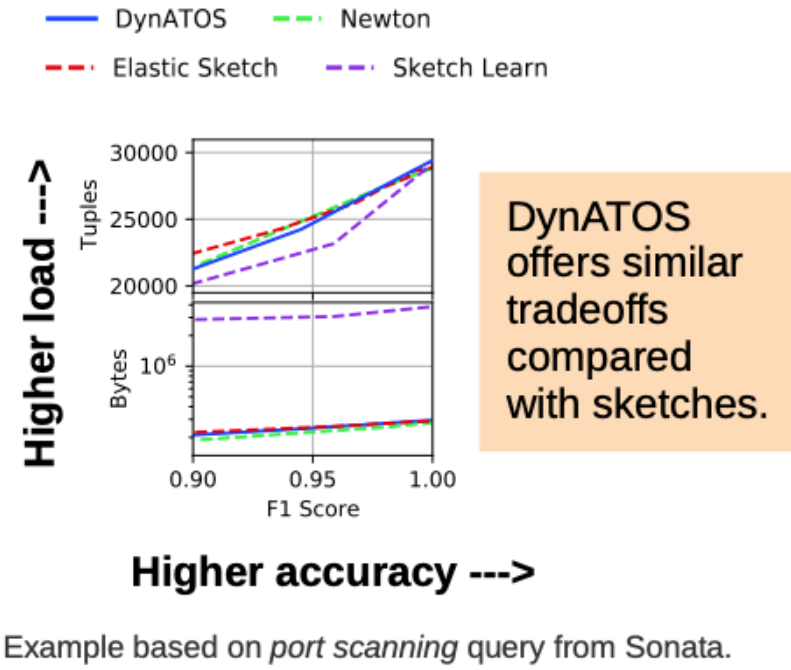
Current limitation

- Can't handle simultaneous traffic and queries.
- Only worked on single match-action table.

Build a closed-loop system to schedule dynamic query workloads on switch hardware.



cmisa@cs.uoregon.edu



Example based on *port scanning* query from Sonata.

Tiara: A Scalable and Efficient Hardware Acceleration Architecture for Stateful Layer-4 Load Balancing

Aims to support high traffic rate (> 1 Tbps), large number of concurrent flows (> 10M), many new connections per second (> 1M) without any assumption on traffic patterns

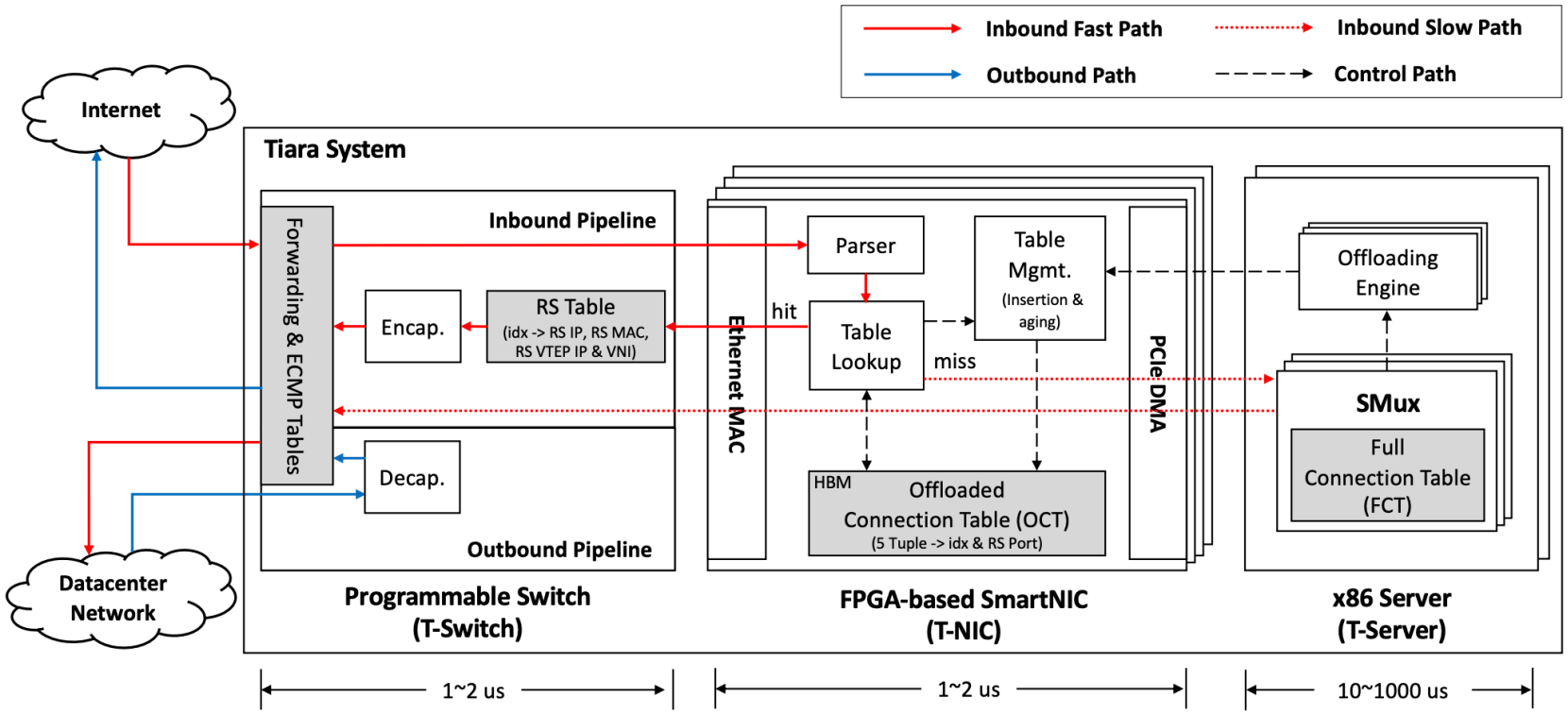


Figure 4: Tiara architecture. Tiara consists of three tiers: T-switch, T-NIC, and T-server. Tiara divides LB into multiple key tasks and matches them respectively to suitable hardware tiers: T-switch for stateless packet encap/decap, T-NIC with HBM for connection lookup and management, and T-server as a last resort.

Notes

Key idea of Tiara is to divide the LB fast path into a memory-intensive task (real server selection) and a throughput-intensive task (packet encap/decap) and map them to suitable hardware, respectively. This is relevant to something we may need to do in our project and could potentially serve as a reference for us.

4. Other interesting papers

Raising the Bar for Programmable Hardware track

- Elixir: A High-performance and Low-cost Approach to Managing Hardware/Software Hybrid Flow Tables Considering Flow Burstiness

  Note: Efficient hardware/software hybrid flow table approach. Could be a good reference for our project(s).

- Gearbox: A Hierarchical Packet Scheduler for Approximate Weighted Fair Queuing

Operational Track

- Decentralized cloud wide-area network traffic engineering with BLASTSHIELD

  Note: Microsoft's software-defined decentralized WAN traffic engineering system.

- Bluebird: High-performance SDN for Bare-metal Cloud Services

  Note: Microsoft's network virtualization system for the bare-metal cloud service in Azure.

- Evolvable Network Telemetry at Facebook

  Note: FB's PCAT, a production change-aware telemetry system.

ISPs and CDNs track

- cISP: A Speed-of-Light Internet Service Provider

  Note: An interesting ISP design using free-space microwave wireless connectivity.

4. Other interesting papers -- continued

Network Troubleshooting & Debug Track

- Collie: Finding Performance Anomalies in RDMA Subsystems
  Note: Anomaly detection tool for RDMA subsystems. ByteDance

- SCALE: Automatically Finding RFC Compliance Bugs in DNS Nameservers
  Note: RFC compliance error detection in DNS ns implementation. UCLA + Microsoft

Reliable Distributed Systems Track

- Graham: Synchronizing Clocks by Leveraging Local Clock Properties. ***Awarded Best Paper!***
  Note: State of the art clock synchronization mechanism which reduces clock drifting by up to 2000x!

Testing and Verification Track

- Performance Interfaces for Network Functions. *George Candea, EPFL*
- Automated Verification of Network Function Binaries. *George Candea, EPFL*
  Note: This is our noticed area since 4 years ago.