

# 使用P4描述复杂网络功能

潘恬

pan@bupt.edu.cn

# 提 纲

第一部分 对P4的认识

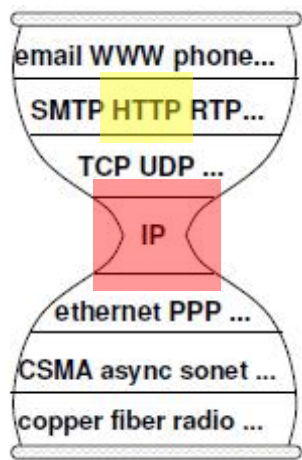
第二部分 P4描述复杂网络功能存在的挑战

第三部分 拓展P4语义支持异步复杂网络功能调用

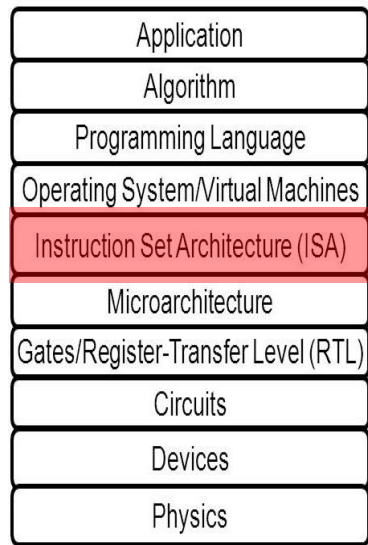
第四部分 P4学术界研究方向

# P4: 数据平面的高级编程语言

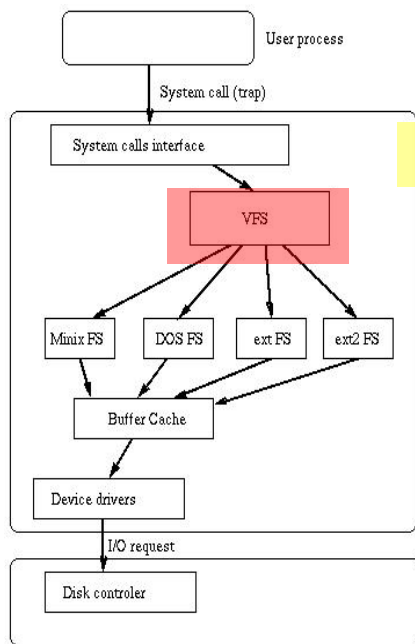
## ■ 计算机系统处处皆是抽象



IP is the thin waist

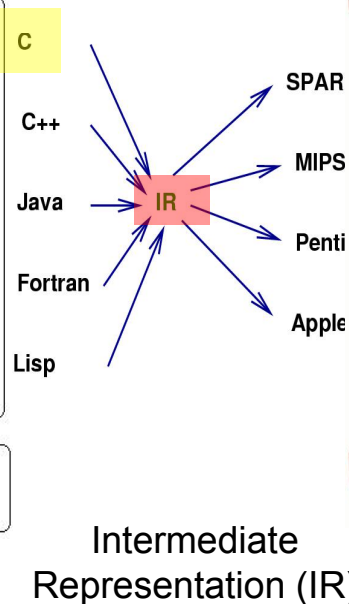


ISA defines CS ARCH

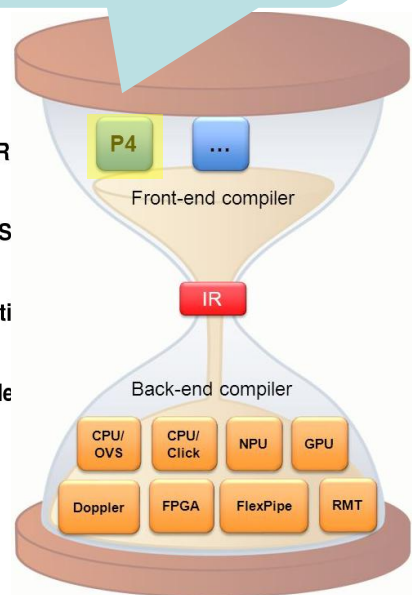


Everything is a file  
in UNIX

Open Question: 未来P4  
是否有可能成为数据平面  
编程语言的一个瘦腰?

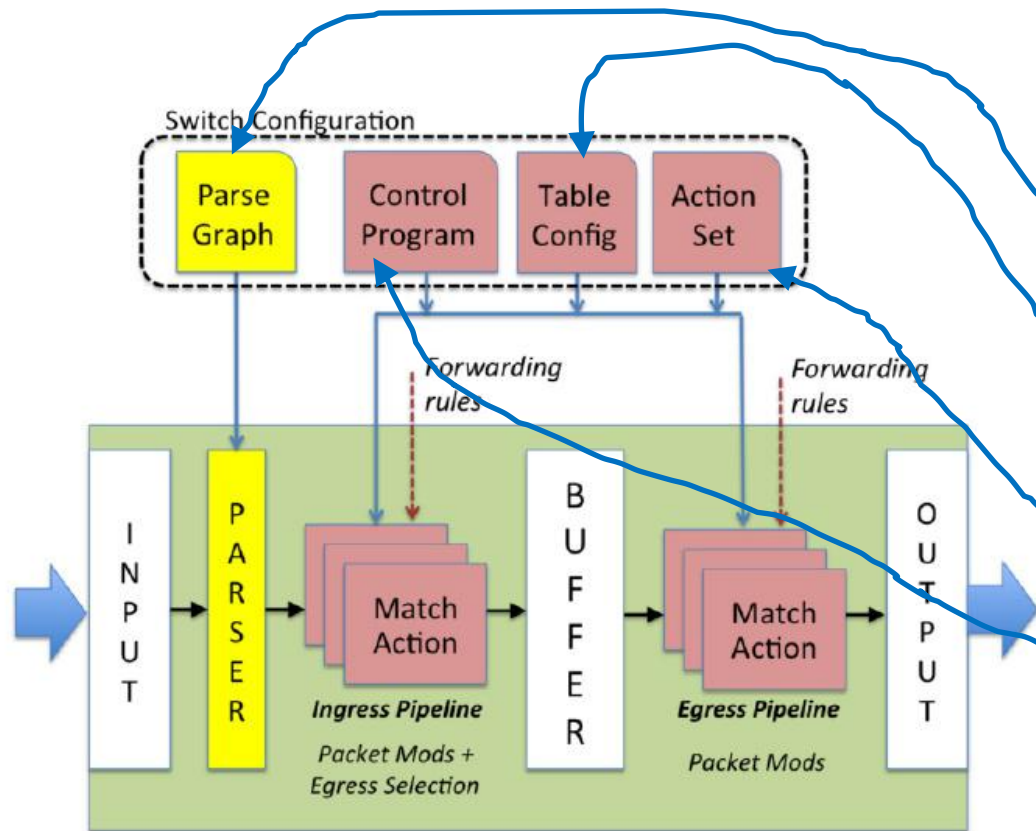


Intermediate  
Representation (IR)



Potential data plane  
language

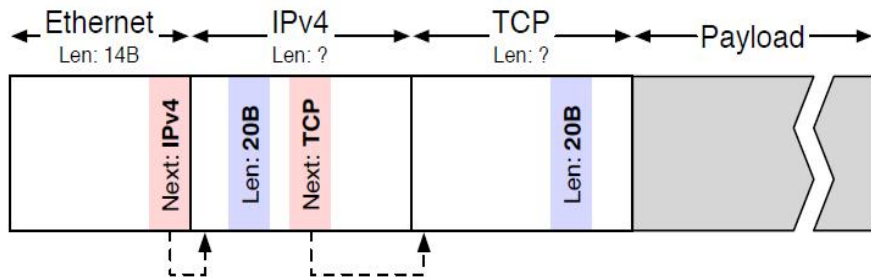
# P4是如何对网络行为进行抽象的？体系结构



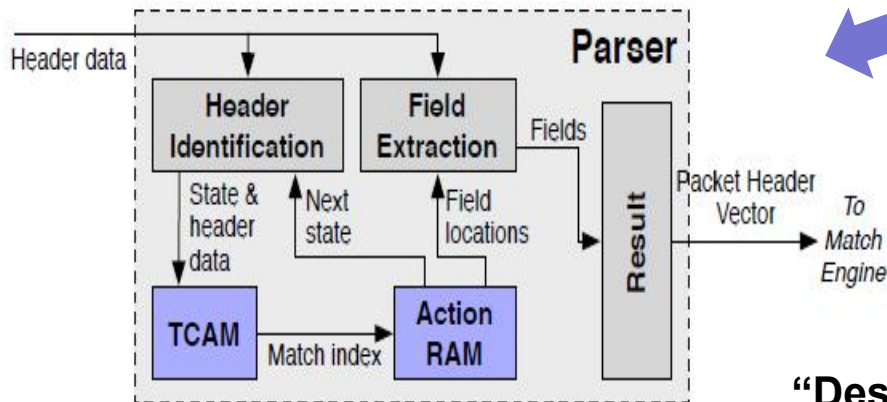
由用户定义以下网络转发行为：

- 协议栈和协议解析方式
- 匹配表数据结构
- 匹配成功或失败后的动作
- 多表之间的数据流走向关系

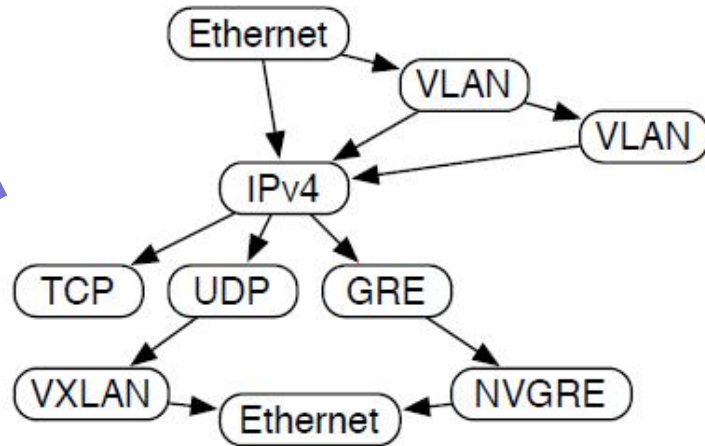
# P4是如何对网络行为进行抽象的？协议解析



包头中的协议栈



协议解析模块的硬件结构

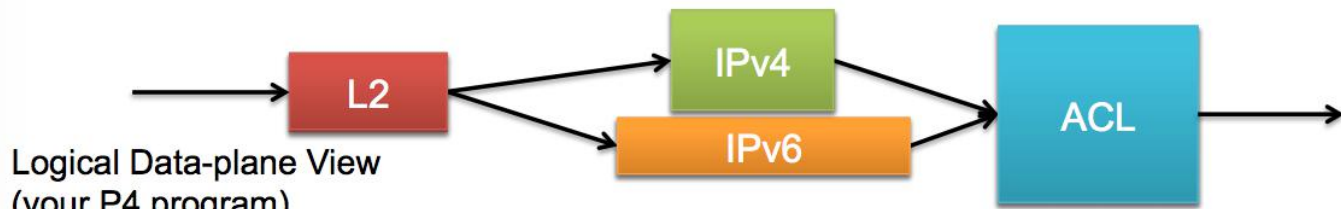


典型数据中心协议的解析过程

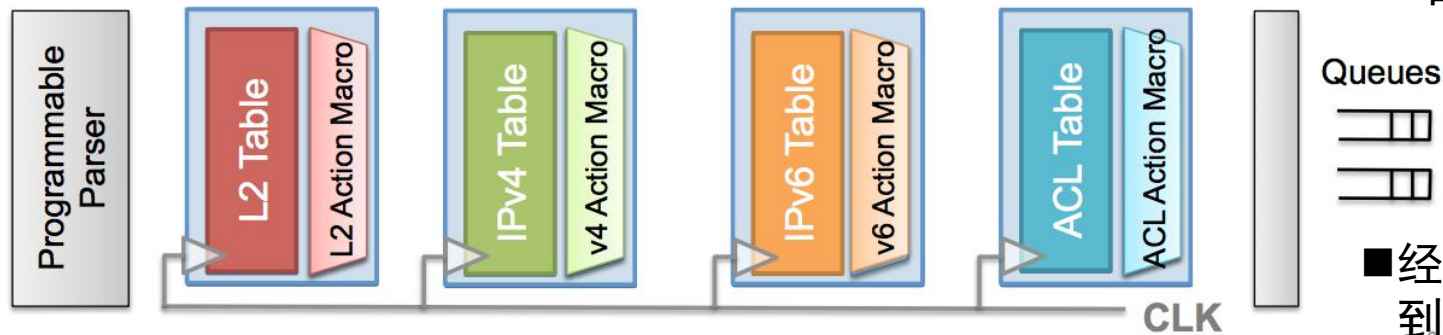
可以灵活构造几乎任意协议栈

“Design Principles for Packet Parsers”, ANCS 2013

# P4是如何对网络行为进行抽象的？查表控制



Switch Pipeline

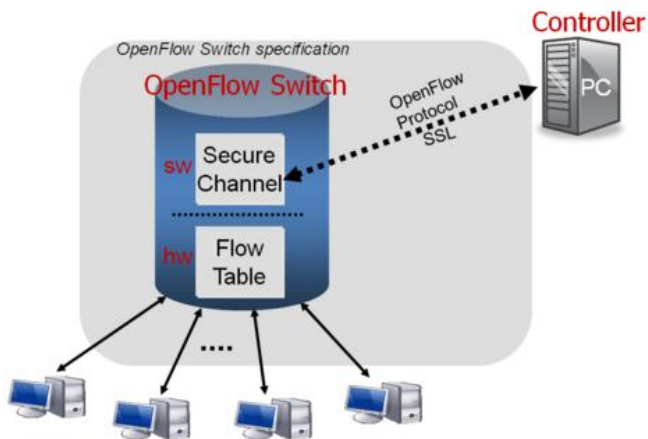


■ 基于有向图描述多表数据流的逻辑关系

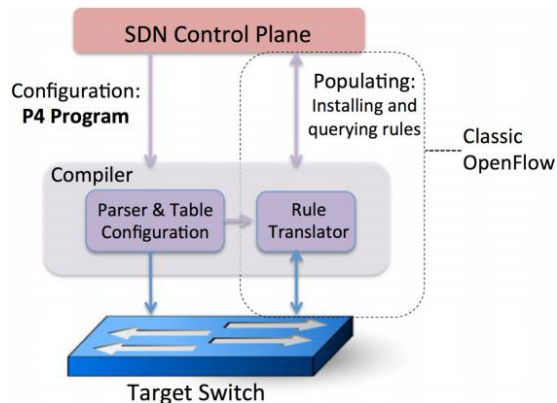
■ 经过编译器映射到支持多表转发的底层交换设备

可以灵活表达多匹配表之间的数据流关系

# P4 vs OpenFlow



演进



- 贡献：
  - 控制与转发分离
  - 流量转发抽象（流表）
- 问题：匹配协议集合固定，版本更迭对硬件厂商不友好

- 对OpenFlow的改进：
  - 不对匹配协议集合作过多假设
  - 不关心具体策略，只提供实现机制

“Separation of mechanism and policy”

# P4 vs PoF

## P4 (Sigcomm 2014 CCR)

```
header ethernet {  
  fields {  
    段  
    dst_addr : 48;  
    src_addr : 48;  
    ethertype : 16;  
  }  
}
```

#头ethernet  
#用fields存储各字段  
#目的地址: 48bit  
#源地址: 48bit  
#以太类型: 16bit

更像高级语言中的结构体

Front-end compiler

IR

Back-end compiler

CPU

NPU

FPGA

ASIC

GPU

## POF (HotSDN 2013)

类型, 偏移地址, 长度{type, offset, length}表示

```
dst: {0, 0, 48};    #起始位0bit, 字段长度48bit  
src: {0, 48, 48};   #起始位48bit, 字段长度48bit  
type: {0, 96, 16};  #起始位96bit, 字段长度16bit
```

更像汇编语言中的偏移寻址

Assembler

FIS

Emulator

FIS executed by POF-enabled machine

2015年之后似乎不太活跃



# 提 纲

第一部分 对P4的认识

第二部分 P4描述复杂网络功能存在的挑战

第三部分 拓展P4语义支持异步复杂网络功能调用

第四部分 P4学术界研究方向

# 网络数据平面功能：无状态转发 & 带状态转发

## 无状态转发

## 带状态转发

### 缘由

中间简单，边缘复杂  
“End-to-end principle”

安全、测量、内容、  
设备辅助流控、高效数据结构等

### 特征

只做尽力而为转发，不维护  
额外状态信息，数据包的转  
发不影响数据平面状态

维护两个包之间的状态信息  
或其他额外状态信息，数据  
包的转发有可能影响数据平  
面状态

# 一些带状态转发的数据平面算法

Algorithm	Stateful operations
Bloom filter (3 hash functions)	Test/Set membership bit on every packet.
Heavy Hitters [63] (3 hash functions)	Increment Count-Min Sketch [31] on every packet.
Flowlets [57]	Update saved next hop if flowlet threshold is exceeded.
RCP [60]	Accumulate RTT sum if RTT is under maximum allowable RTT.
Sampled NetFlow [17]	Sample a packet if packet count reaches N; Reset count to 0 when it reaches N.
HULL [22]	Update counter for virtual queue.
Adaptive Virtual Queue [47]	Update virtual queue size and virtual capacity
Priority computation for weighted fair queueing [58]	Compute packet's virtual start time using finish time of last packet in that flow.
DNS TTL change tracking [26]	Track number of changes in announced TTL for each domain
CONGA [21]	Update best path's utilization/id if we see a better path. Update best path utilization alone if it changes.
CoDel [51]	Update: Whether we are marking or not. Time for next mark. Number of marks so far. Time at which min. queueing delay will exceed target.

摘自 “Packet Transactions: High-Level Programming for Line-Rate Switches” , SIGCOMM 2016

## ■ 带状态转发对计算和存储的要求

- 大容量存储单元 (BF、CCN)
- 判断并分支 (Flowlets)
- 浮点比较 (WFQ)
- 计时器老化操作 (DPI)
- 复杂运算 (DPI)
- 取得队列参数 (Queuing Algorithms)

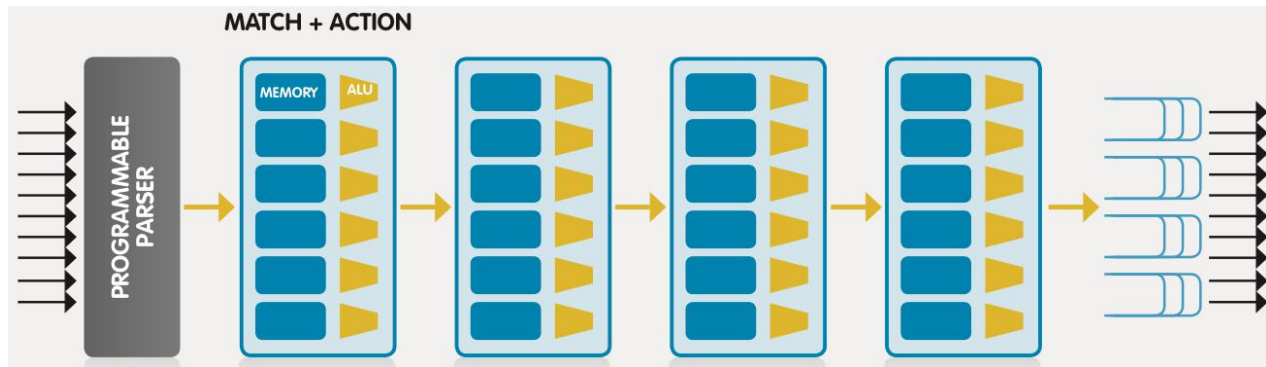
尽管带状态转发算法破坏了“端到端”原则，但考虑到其在流控、安全、调度等方面获得的性能收益，仍然被广泛推广使用，尤其近年来数据中心网络中的大量创新都集中于此

# P4能够轻松描述无状态转发

```
table routing {  
  reads {  
    ipv4.dstAddr : lpm;  
  }  
  actions {  
    do_drop;  
    route_ipv4;  
  }  
  size: 2048;  
}  
  
control ingress {  
  apply(routing);  
}
```

## ■ 无状态转发的主要操作：

- 查表得到相应动作
- 动作一般异常简单，如丢弃、转发等
- 简单的包头字段的更新（如TTL）
- 不需要访问数据平面额外状态（如队列等）
- 操作基本都能在一个流水段内快速完成



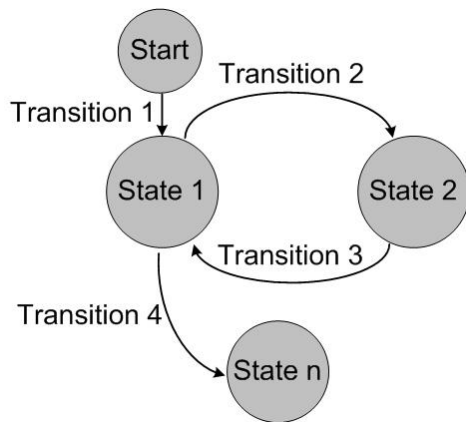
# P4也能够描述一部分带状态转发功能

## 8 Counters, Meters and Registers

Counters, meters and registers maintain state for longer than one packet. Together they are called stateful memories. They require resources on the target and hence are managed by a compiler.

From P4 Spec V1.1.0

- PISA架构基于多流表，因此能够维护每流的状态
- Counter，Meter和Register的操作比较简单，不会带来过长时延，基本的操作有load、store、ALU、etc
- 可以将状态机FSM维护在Register中，从而进行跨包状态跳转的监测，例如识别TCP的三次握手



# P4是否已足够强大支持所有带状态转发功能？

世界上没有一劳永逸的万能方法

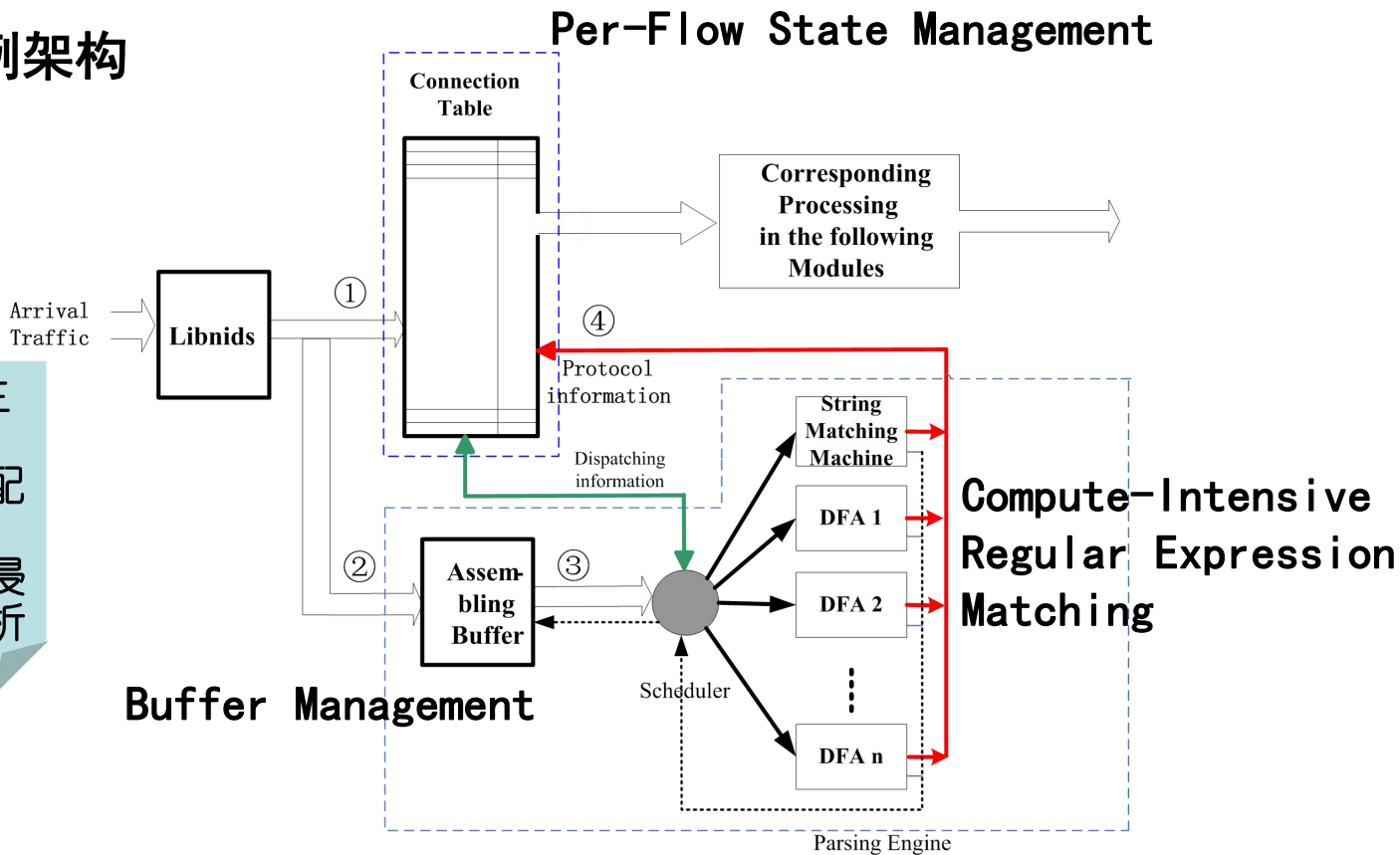
**NO**



# Case Study: 深度包处理 (DPI)

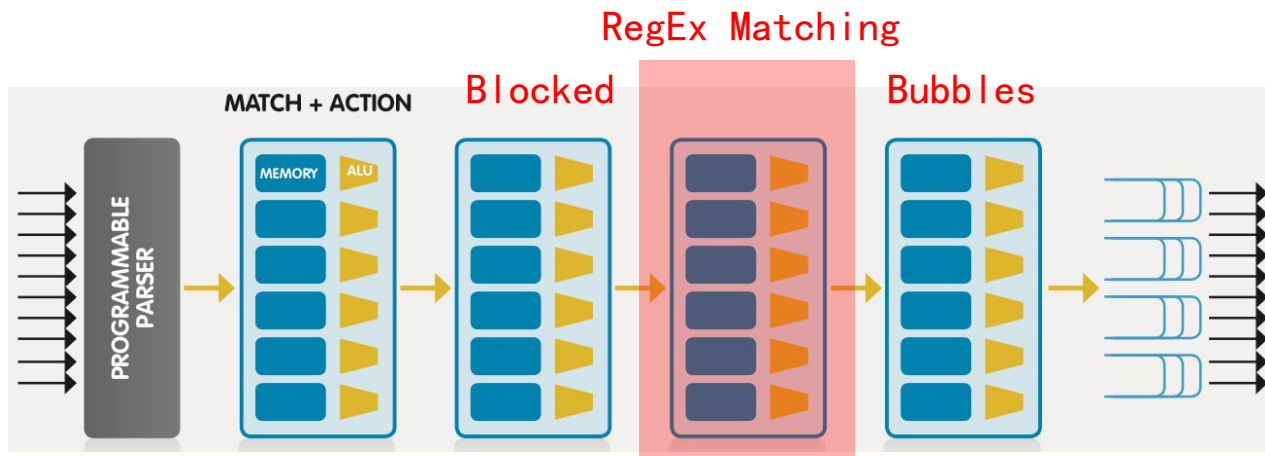
## 一个DPI的示例架构

深度包处理 (DPI) 主要通过文法描述规则 (如正则表达式) 匹配数据包应用层的内容, 主要用于防火墙、入侵检测系统、大数据分析等领域



# P4描述DPI遇到的问题

- 大时延操作导致流水线阻塞和气泡
- 无法更新数据平面状态（如插入流表）
- 无法有效访问数据平面（如数据包payload）
- 无法有效的进行缓存管理（设备对编程者透明）





# P4的Primitive Actions主要关注包头处理

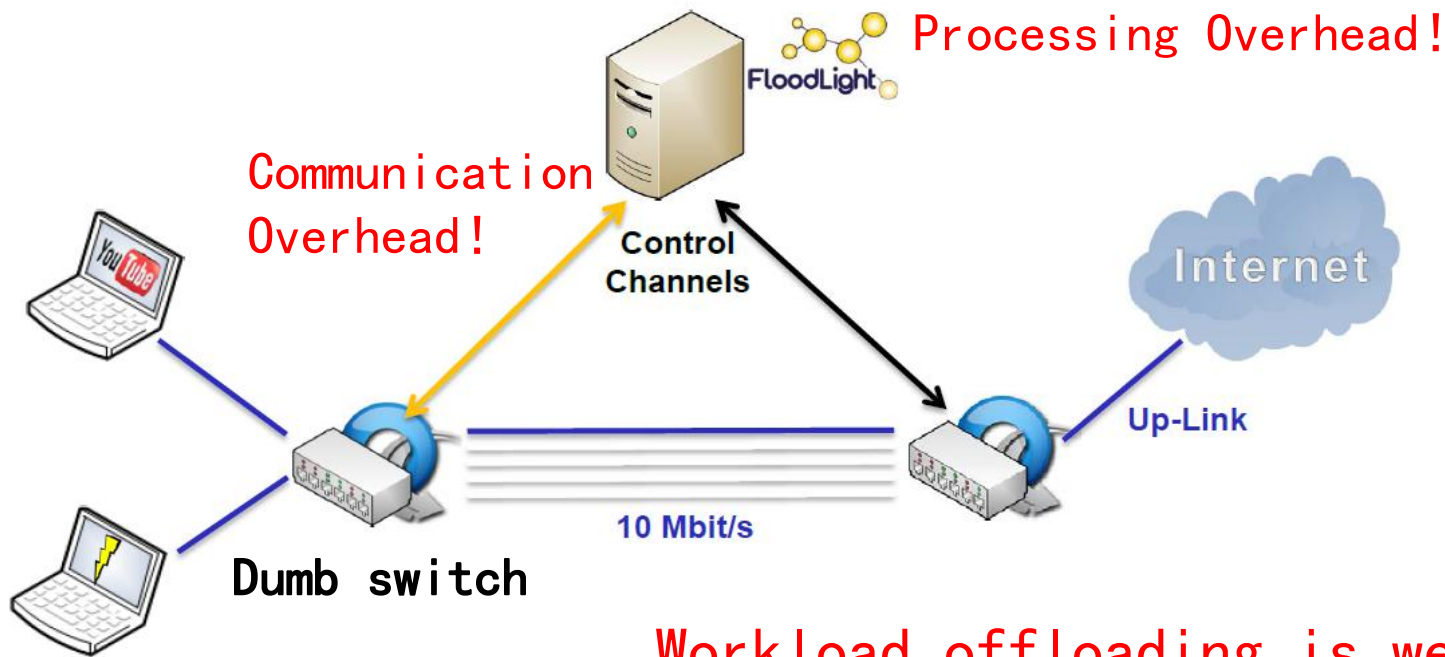
primitive name	Summary
add_header	Add a header to the packet's Parsed Representation
copy_header	Copy one header instance to another.
remove_header	Mark a header instance as invalid.
modify_field	Set the value of a field in the packet's Parsed Representation.
modify_field_with_hash_based_offset	Apply a field list calculation and use the result to generate an offset value.
truncate	Truncate the packet on egress.
drop	Drop a packet (in the egress pipeline).
no_op	Placeholder action with no effect.
push	Push all header instances in an array down and add a new header at the top.
pop	Pop header instances from the top of an array, moving all subsequent array elements up.

count	Update a counter.
meter	Execute a meter operation.
generate_digest	Generate a packet digest and send to a receiver.
resubmit	Resubmit the original packet to the parser with metadata.
recirculate	Resubmit the packet after all egress modifications.
clone_ingress_pkt_to_ingress	Send a copy of the original packet to the parser. Alias: clone_i2i.
clone_egress_pkt_to_ingress	Send a copy of the egress packet to the parser. Alias: clone_e2i.
clone_ingress_pkt_to_egress	Send a copy of the original packet to the Buffer Mechanism. Alias: clone_i2e.
clone_egress_pkt_to_egress	Send a copy of the egress packet to the Buffer Mechanism. Alias: clone_e2e.

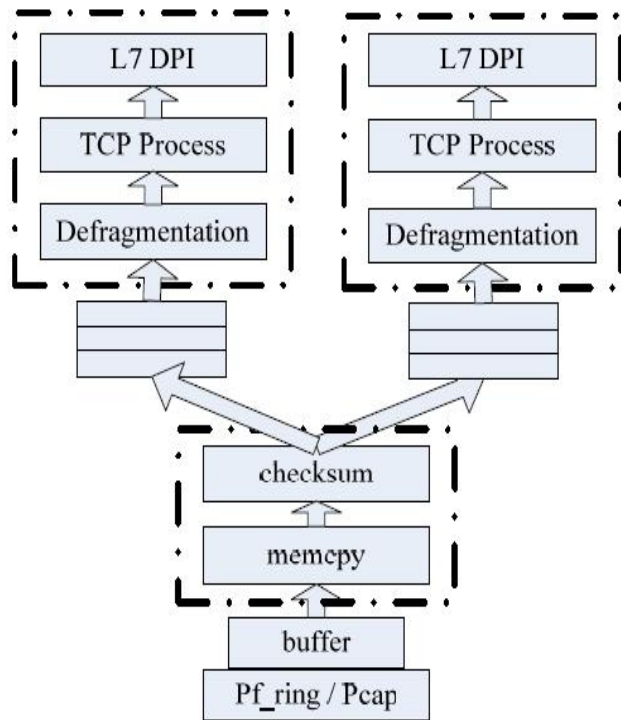
然而，一些数据平面算法要求更新数据平面状态（例如插入流表）

# OpenFlow如何处理DPI?

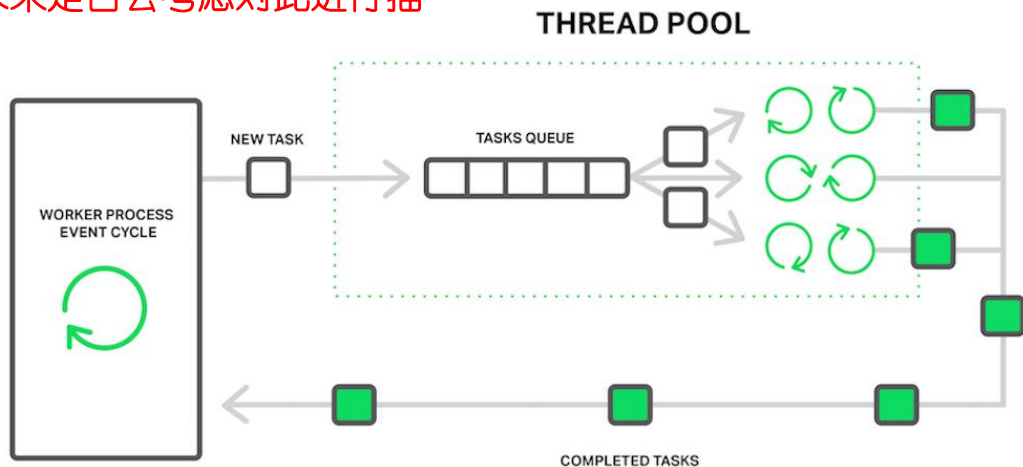
DPI in controller or middle boxes



# 网络处理器或x86平台如何处理DPI?



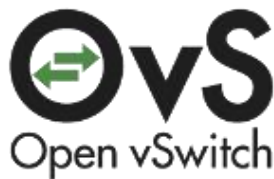
这种处理方式和**P4**默认的流水线硬件具有较大差别，**P4**语言作为设备无关的数据平面高级编程语言未来是否会考虑对此进行描述？



数据平面多核的每个子线程采用run-to-complete的调度方式并发处理到达的任务

# P4在边缘网络的潜在竞争对手

- P4足够胜任高速网络领域
- C语言比P4有着更强的表达能力
- 但C语言面向的设备处理能力较弱
- 但在边缘网络，这种劣势被削弱了



# 提 纲

第一部分 对P4的认识

第二部分 P4描述复杂网络功能存在的挑战

第三部分 拓展P4语义支持异步复杂网络功能调用

第四部分 P4学术界研究方向

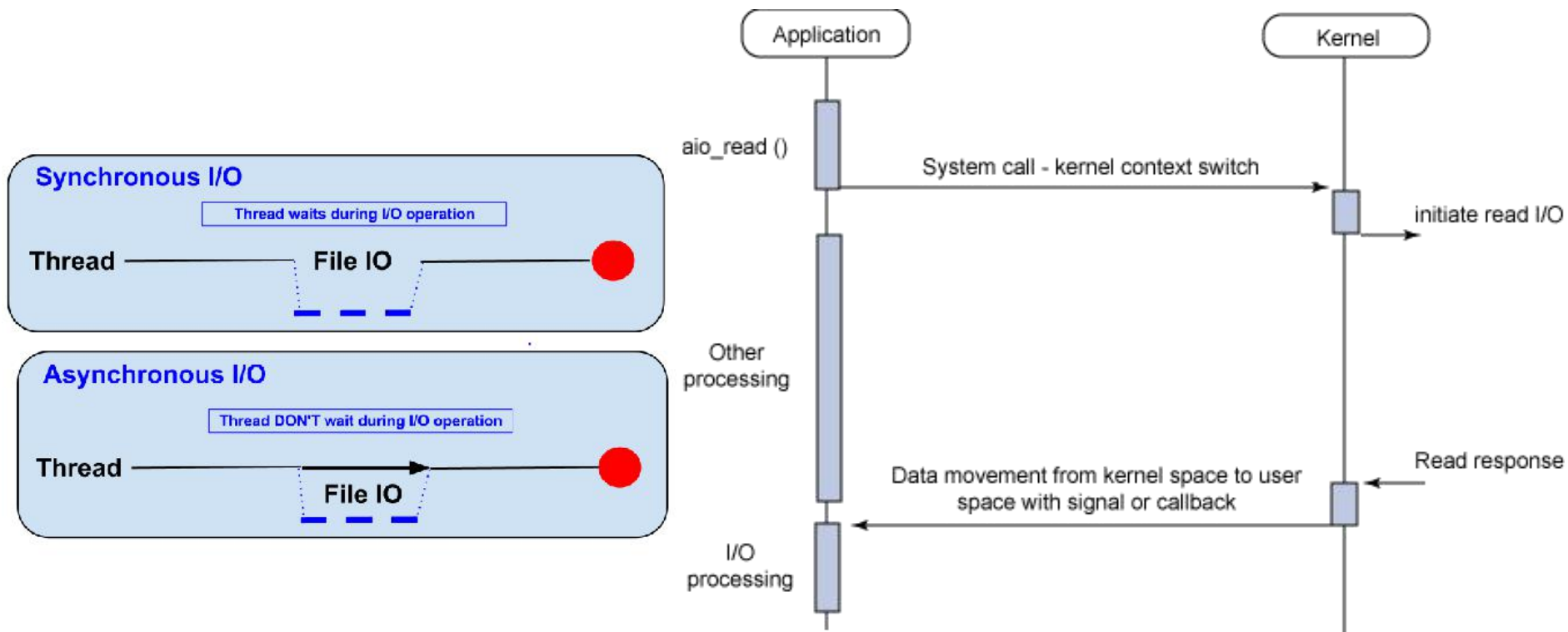
# 针对复杂网络功能的P4抽象机制扩展

## ■ 数据包的不同行为需要不同的描述和处理机制

数据包行为	例子	P4处理机制
查询数据平面状态（快）， 影响自身转发	IPv4 Route Lookup	Pipeline is OK
查询数据平面状态（慢）， 影响自身转发	Off-chip Memory involved, e.g., CCN	Pipeline has to be blocked or splitted
查询数据平面状态， 修改数据平面状态（快）， 影响后续包转发	Most stateful forwarding cases, e.g., Sampled NetFlow	Pipeline is OK if <b>ALU operations are well supported</b>
查询数据平面状态，修改 数据平面状态（慢），影 响后续包转发	DPI	<b>Pipeline bubbles occur, data path is blocked</b>

# 异步无阻塞调用作为一种调用抽象机制

## ■ 典型例子：Non-Blocking I/O



# 描述DPI的P4逻辑片段

```
// P4 functions with async call logic
```

```
// header definitions
```

```
...
```

```
// parser definitions
```

```
...
```

```
// table definitions
```

```
...
```

```
table flow_table{
  reads{
    flow_tuple : exact
    payload : buffer
  }
  actions{
    if (!hit(flow_tuple, flow_table))
      regex_scan(payload, callback);
    forward_with_policy;
  }
}
```

```
// callback function definitions
```

```
void callback (result){
  update_table(flow_table, result);
}
```

```
// control definitions
```

```
...
```

```
// c functions for RegEx matching
```

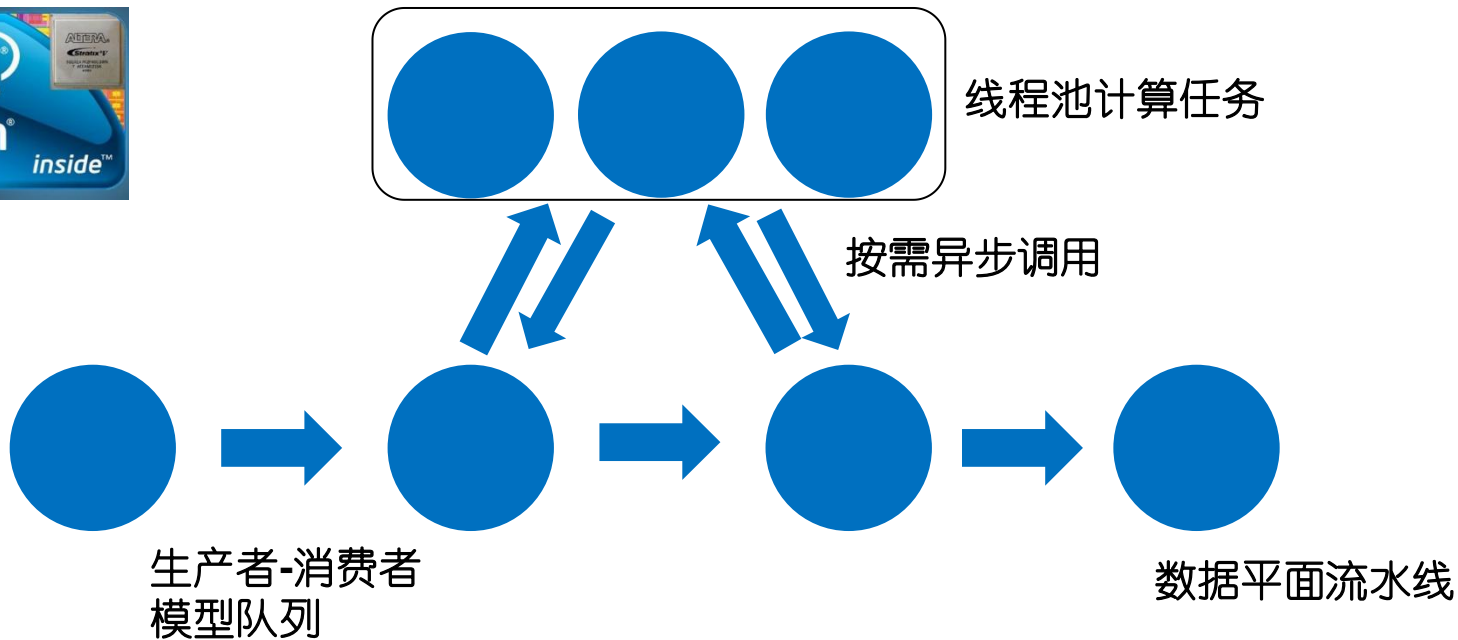
```
void regex_scan (payload, callback){
  ....
}
```

对比目前P4，一些语法上的变化：

1. DPI需要看到数据包的payload
2. 需要查表结果进行分支操作（不确定P4目前是否支持）
3. 异步调用机制及回调函数书写以影响设备状态
4. 与C语言的目标文件进行链接以扩展任意功能

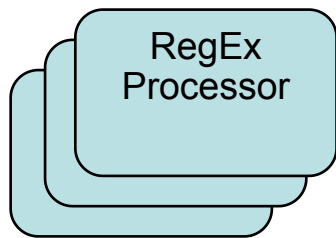


# 硬件支持：x86服务器



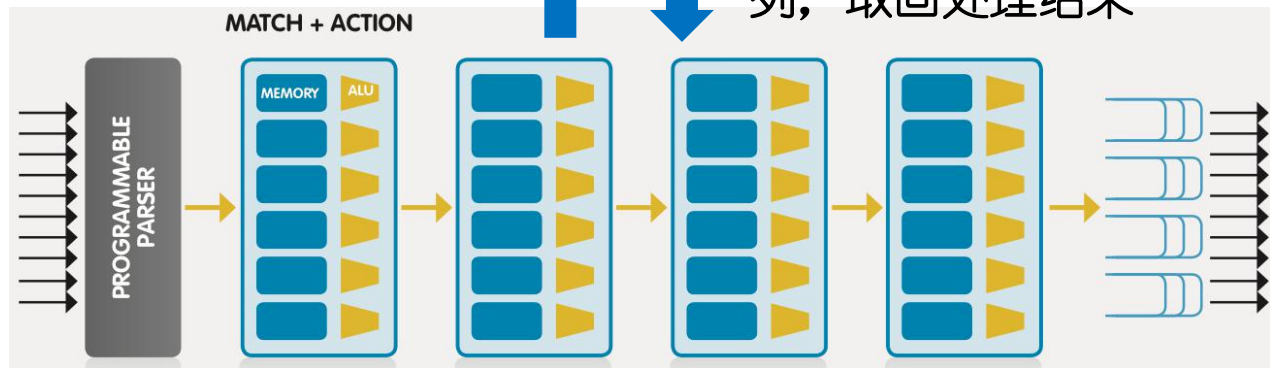
# 硬件支持：交换机硬件+正则表达式协处理器

协处理器堆提供计算能力  
协处理器可以是专用处理器，也可以是通用**RISC**处理器提供分支跳转等基本运算能力



数据流除了在流水线中流动，也有可能按需流入流出协处理器堆

流水线端在每个（或多个）时钟周期定期**Poll**消息队列，取回处理结果



# 提 纲

第一部分 对P4的认识

第二部分 P4描述复杂网络功能存在的挑战

第三部分 拓展P4语义支持异步复杂网络功能调用

第四部分 P4学术界研究方向

# 学术界已开展的P4相关研究课题

## ■ 后端编译器

- GPU

- FPGA

## ■ 混合编程

- P4+OVS

## ■ 数据中心应用

- DC. p4

- Inband network telemetry

- Network verification

- Network measurement

## ■ 带状态转发

- OpenState

- Domino

## ■ 流量调度硬件抽象

- Programmable scheduler

## ■ 数据平面抽象语言

- SAI (Switching Abstractions Interface)


From P4 workshop, Sigcomm, SOSR, HotNets, etc

# 未来其他一些研究方向

- P4语言在数据中心私有协议方面的应用
- P4语言在描述数据平面排队和调度逻辑方面的扩展
- 基于x86处理器对P4语言应用的加速处理机制
- P4语言在数据平面虚拟化切片方面的应用
- P4语言在深度报文处理方面的应用
- P4语言协议无关解析模块的性能优化

# 总结

- P4是一门全新的数据平面高级编程语言
- P4的平台无关特性将使开发人员收益
- P4主要面向高速网络交换，在面向边缘网络带状态处理时尚存挑战
- 提出一种基于异步编程的P4扩展框架，希望能够利用通用编程语言的丰富表达能力对目前的P4文法进行弥补，同时尽量不增加数据包处理时延，该机制也有利于SDN控制器做workload offloading

A thick red horizontal bar spans the top of the slide, with a red square in the top-left corner.

**谢 谢**