A Layered Formal Methods Approach to Answering Queue-related Queries

Divya Raghunathan, Maria Apostolaki, Aarti Gupta

Princeton University

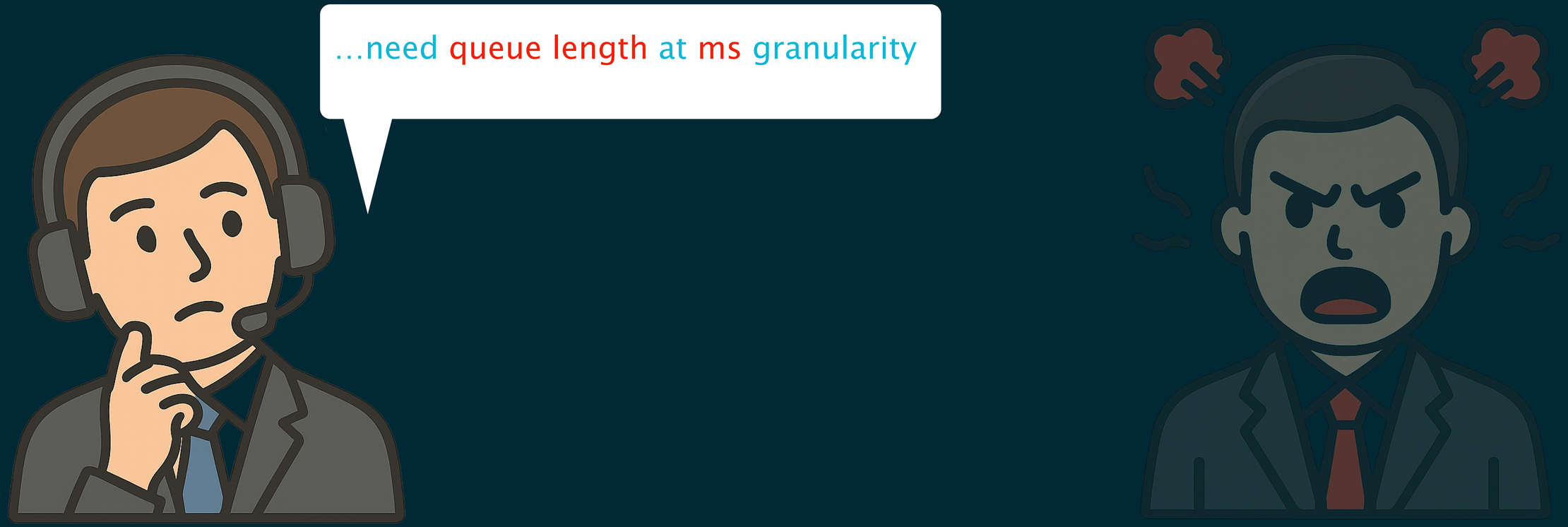
netsyn.princeton.edu

… imagine you are working at a cloud provider

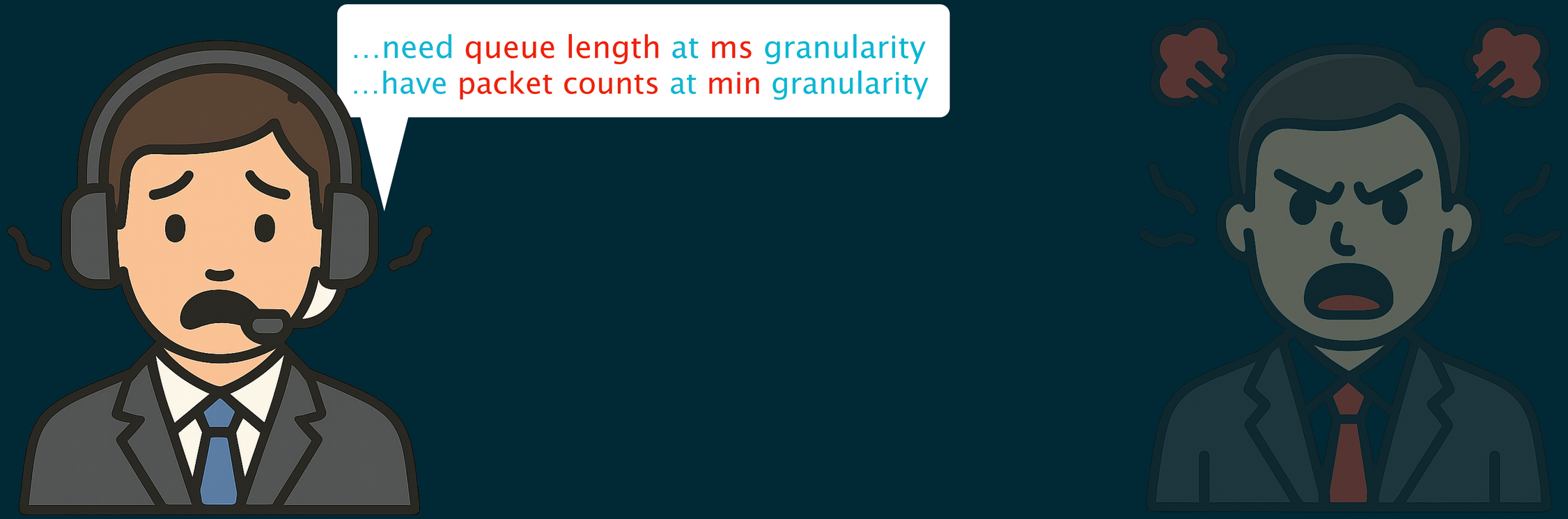
An angry client accuses the cloud of violating a latency SLO

The cloud is only liable, iff the queue length on a port used by the client’s traffic exceeded a threshold, high enough to cause a violation

The cloud is only liable, iff the queue length on a port used by the client's traffic exceeded a threshold, high enough to cause a violation



The cloud is only liable, iff the queue length on a port used by the client's traffic exceeded a threshold, high enough to cause a violation



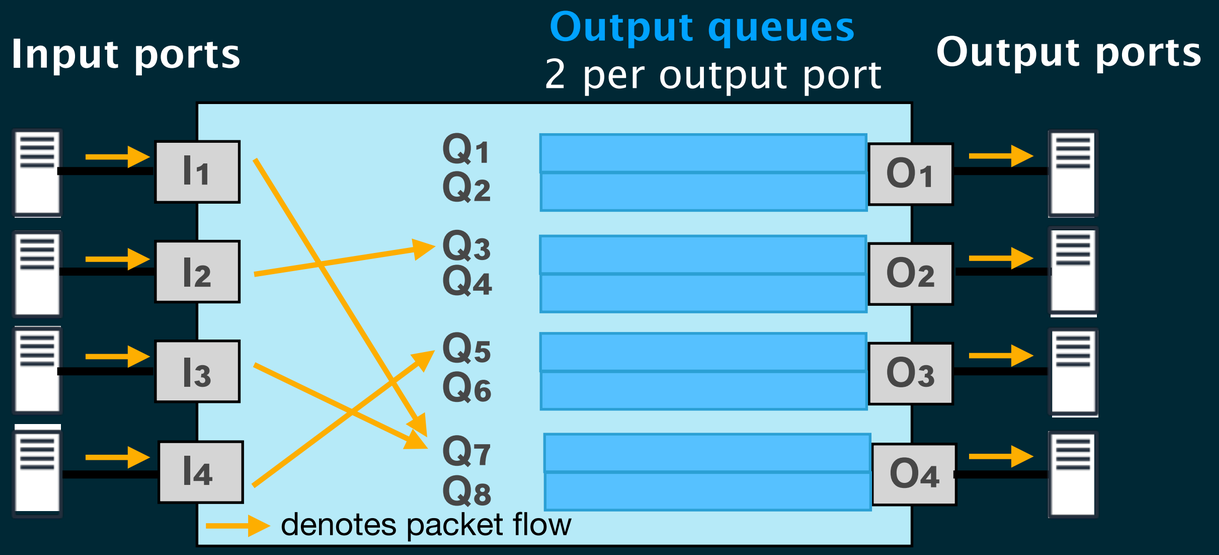
Can one still prove that the latency SLO was not violated,
using only packet counts?

Can one answer queue-related queries,
using only packet counts?

Can one answer queue-related queries,
using only packet counts?

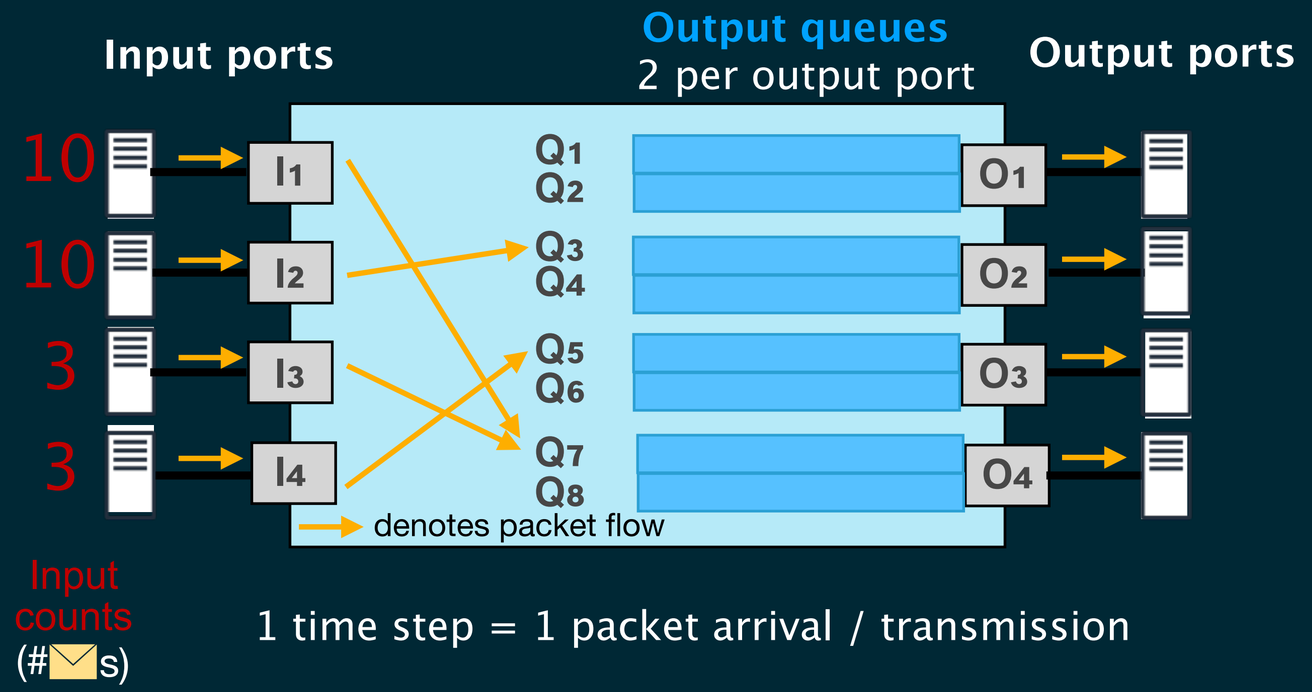
Hint: There is a connection between packet counts and queue lengths

The network operator has access to per-port input, output and dropped packet counts but needs to reason about queue lengths



1 time step = 1 packet arrival / transmission

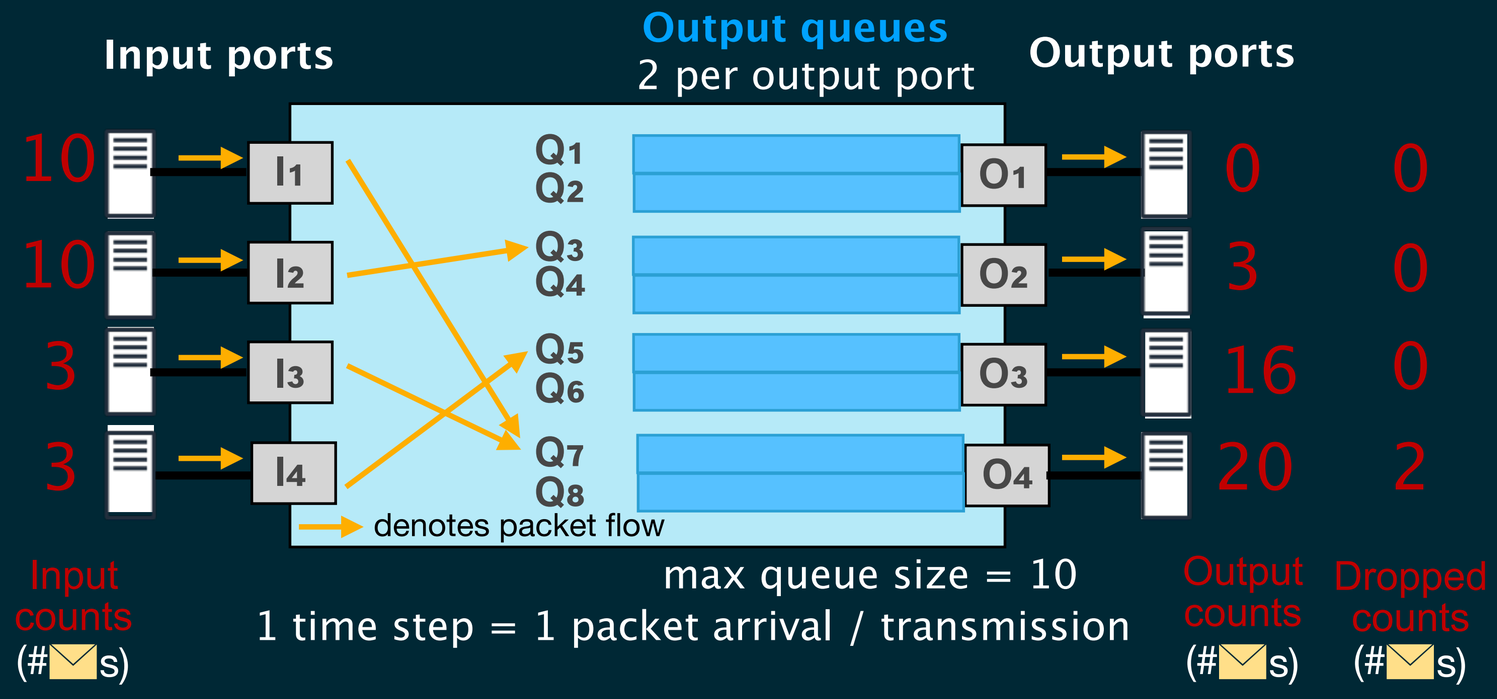
The network operator has access to per-port input, output and dropped packet counts but needs to reason about queue lengths



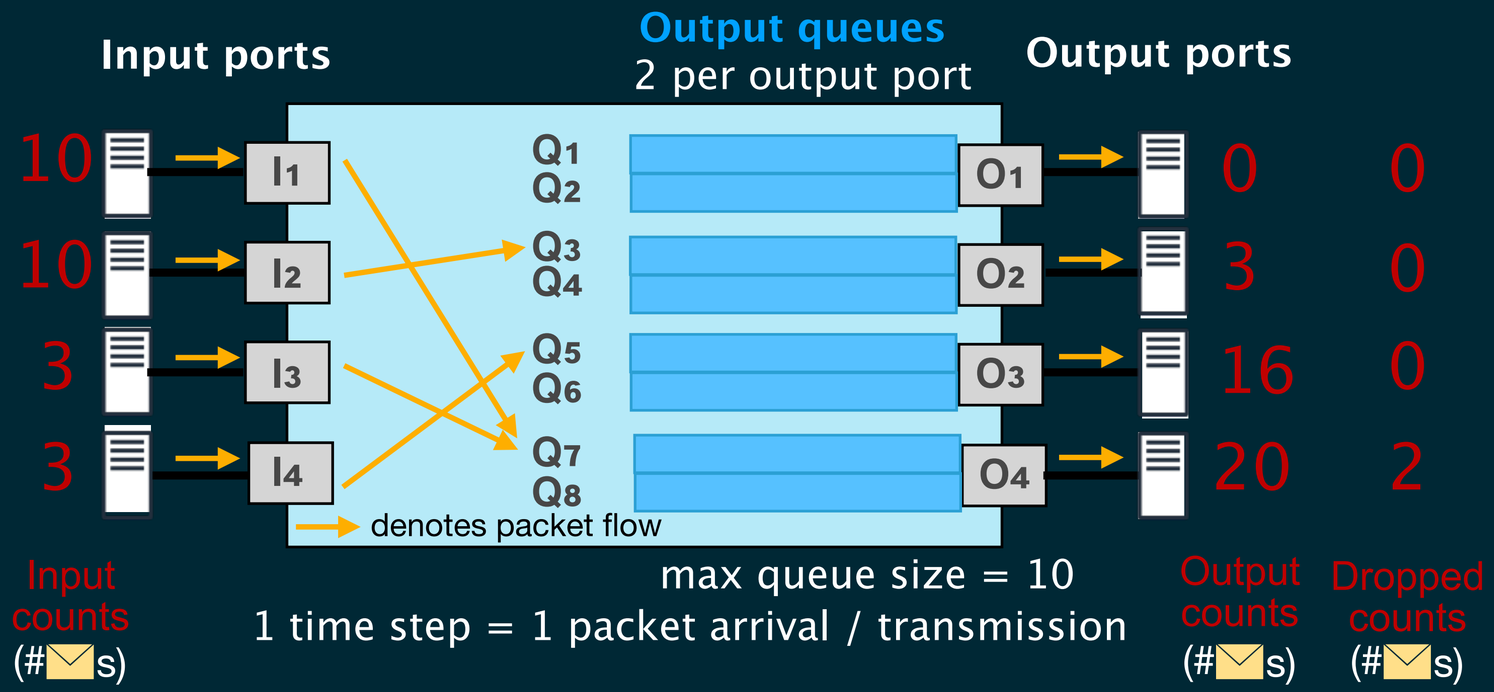
1 time step = 1 packet arrival / transmission

20 time steps

The network operator has access to per-port input, output and dropped packet counts but needs to reason about queue lengths

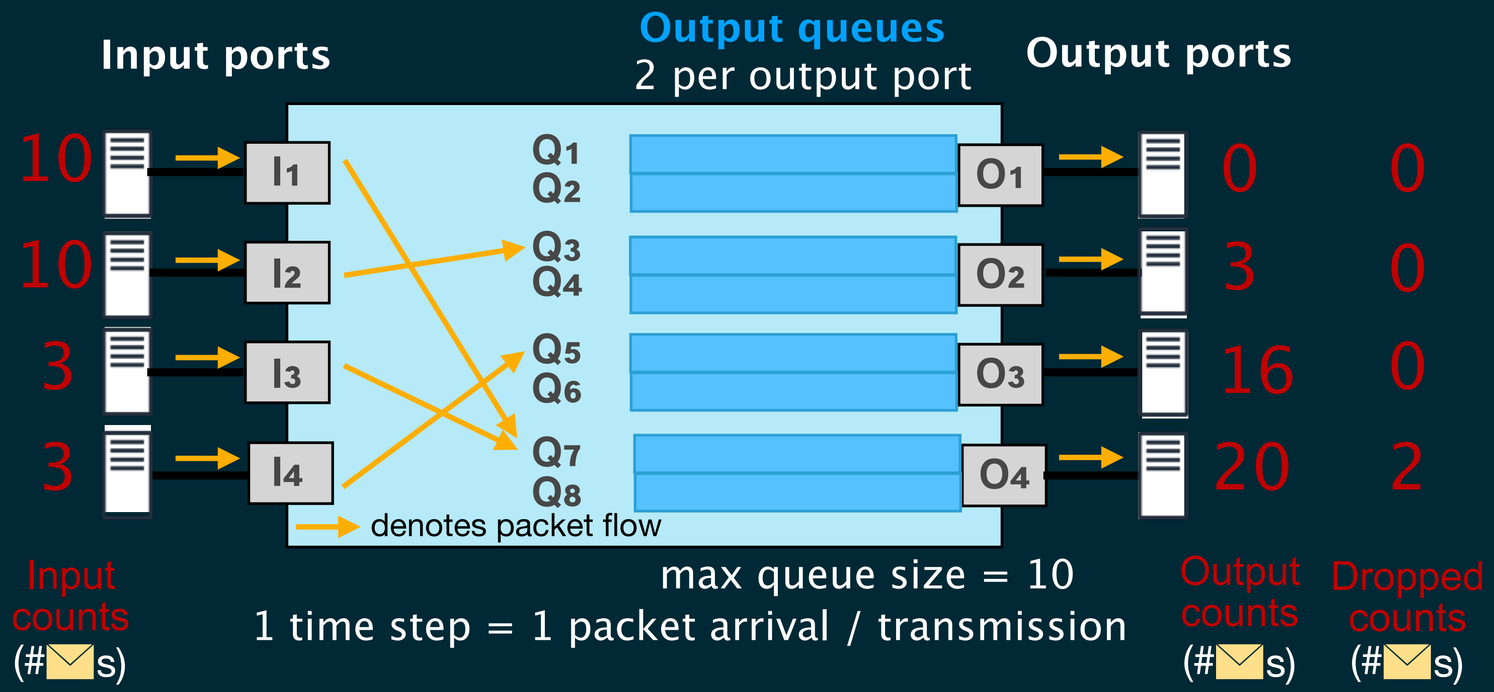


The network operator has access to per-port input, output and dropped packet counts but needs to reason about queue lengths



Can queue size of port O_2 be ≥ 7?

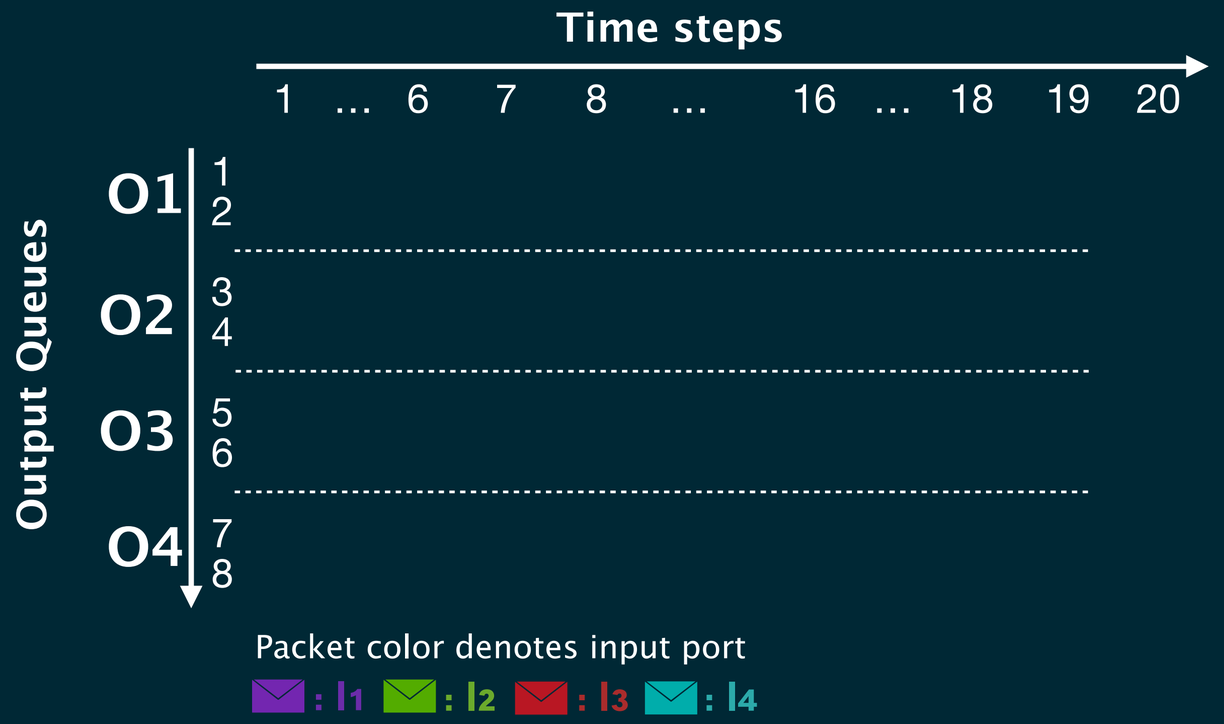
The network operator needs to find a sequence of packet arrivals, that satisfy the packet counts and lead to such a queue length



Can queue size of port O_2 be ≥ 7?

The network operator needs to find a Packet Trace, that satisfy the packet counts and lead to a queue length of 7 pkts

Packet Trace

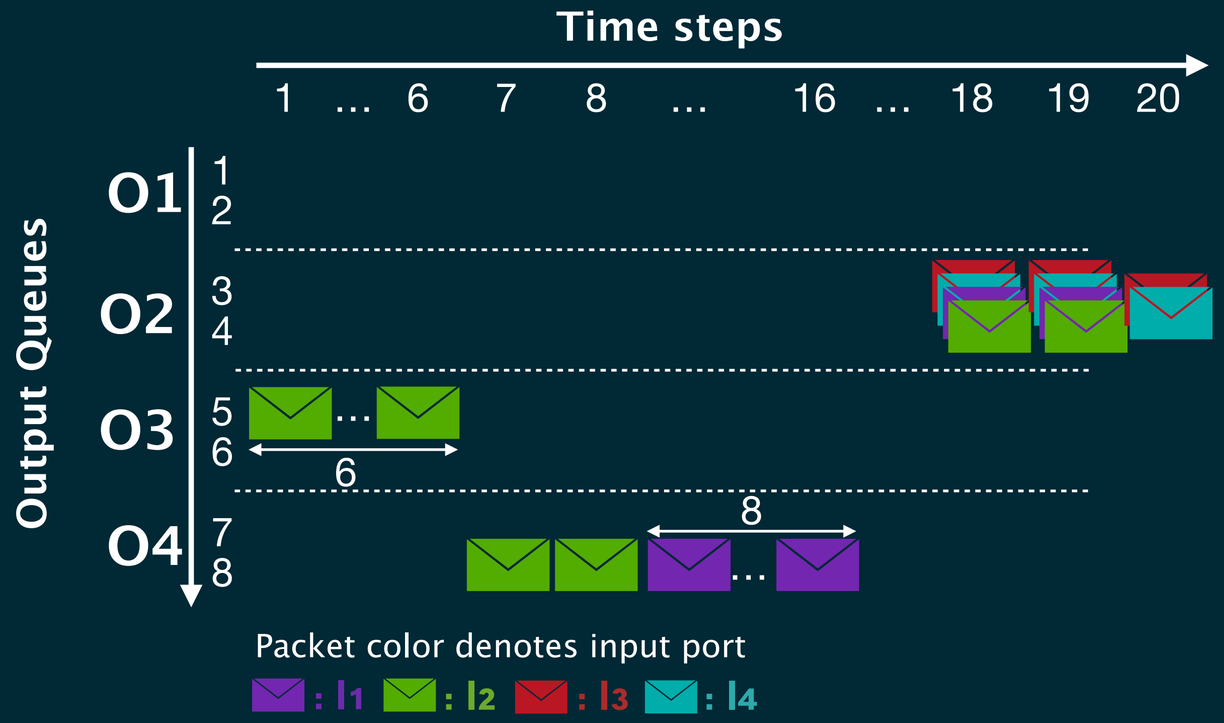


Measurements

Input counts (# s)	Output counts (# s)	Dropped counts (# s)
10	0	0
10	3	0
3	16	0
3	20	2

The network operator needs to find a Packet Trace, that satisfy the packet counts and lead to a queue length of 7 pkts

Packet Trace

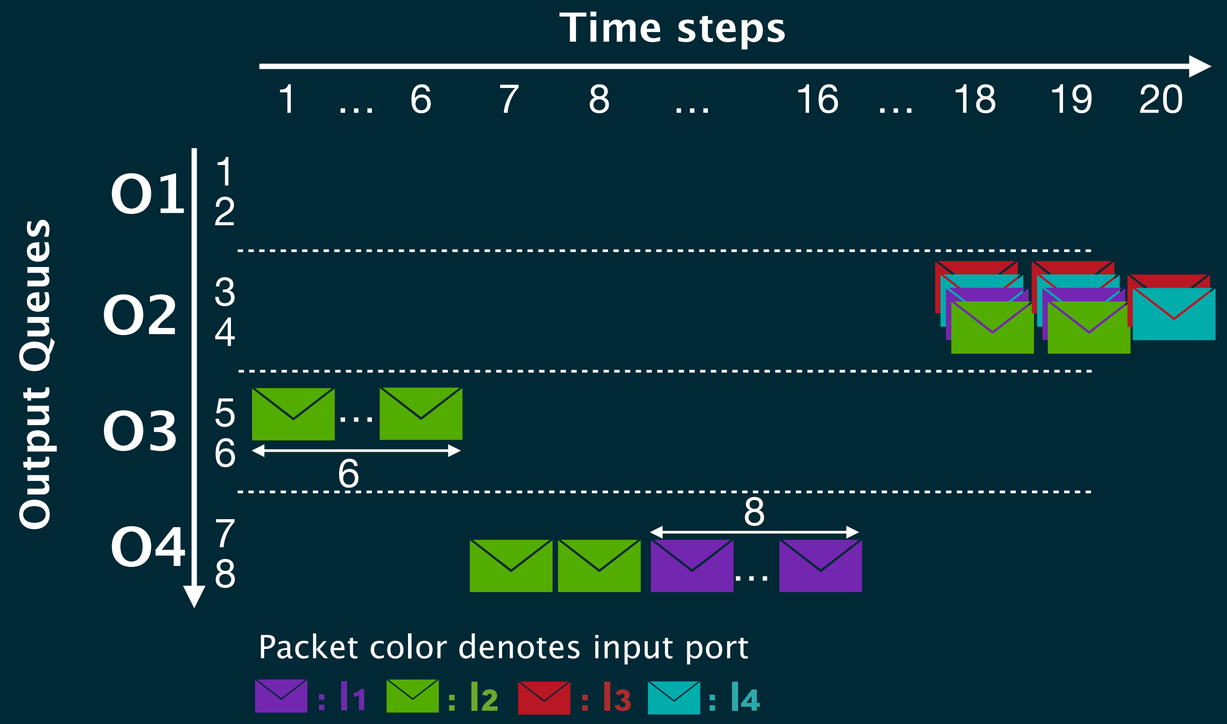


Measurements

Input counts (# s)	Output counts (# s)	Dropped counts (# s)
10		
10		
3		
3		

The operator can start with input counts and generate a packet trace that will satisfy those and the query (O_2> 7 pkts)

Packet Trace



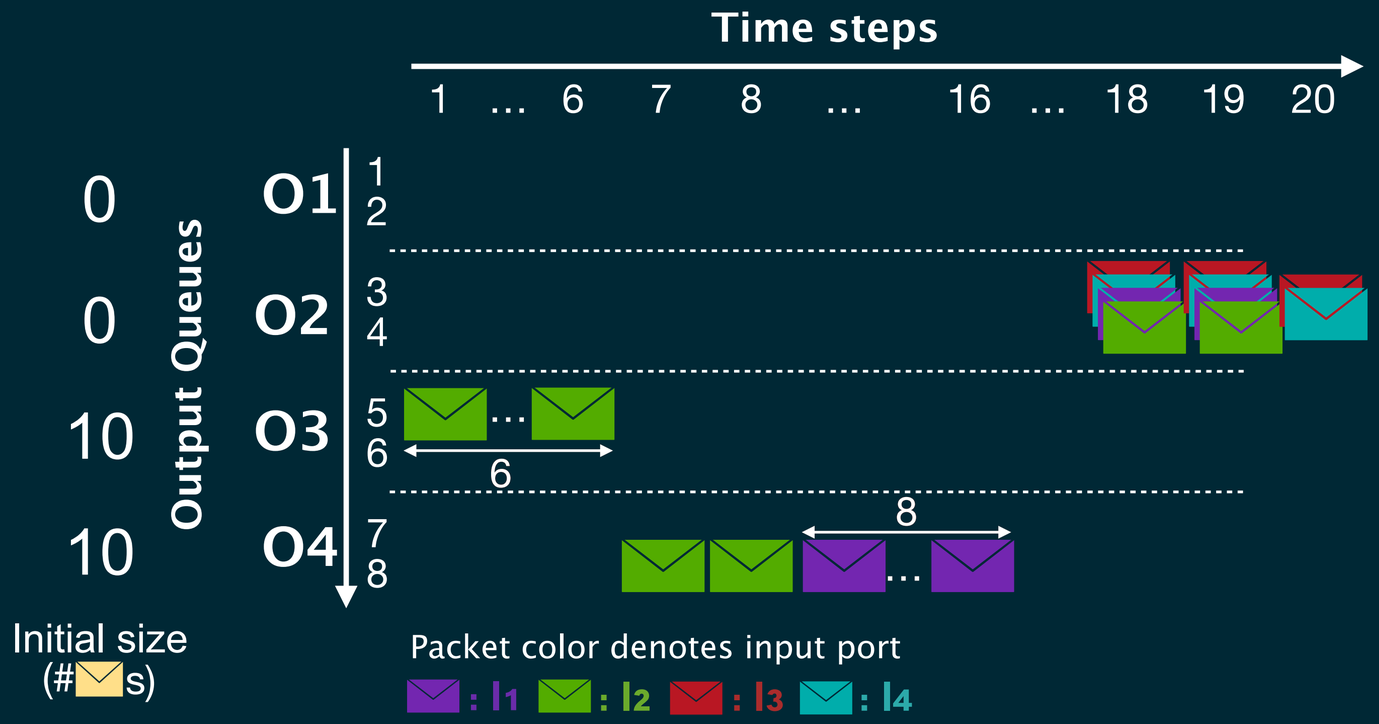
Measurements

Input counts (# s)	Output counts (# s)	Dropped counts (# s)
10	0	
10	3	
3	16	
3	20	

✓

The packet trace is consistent with input, output counts and the query

Packet Trace

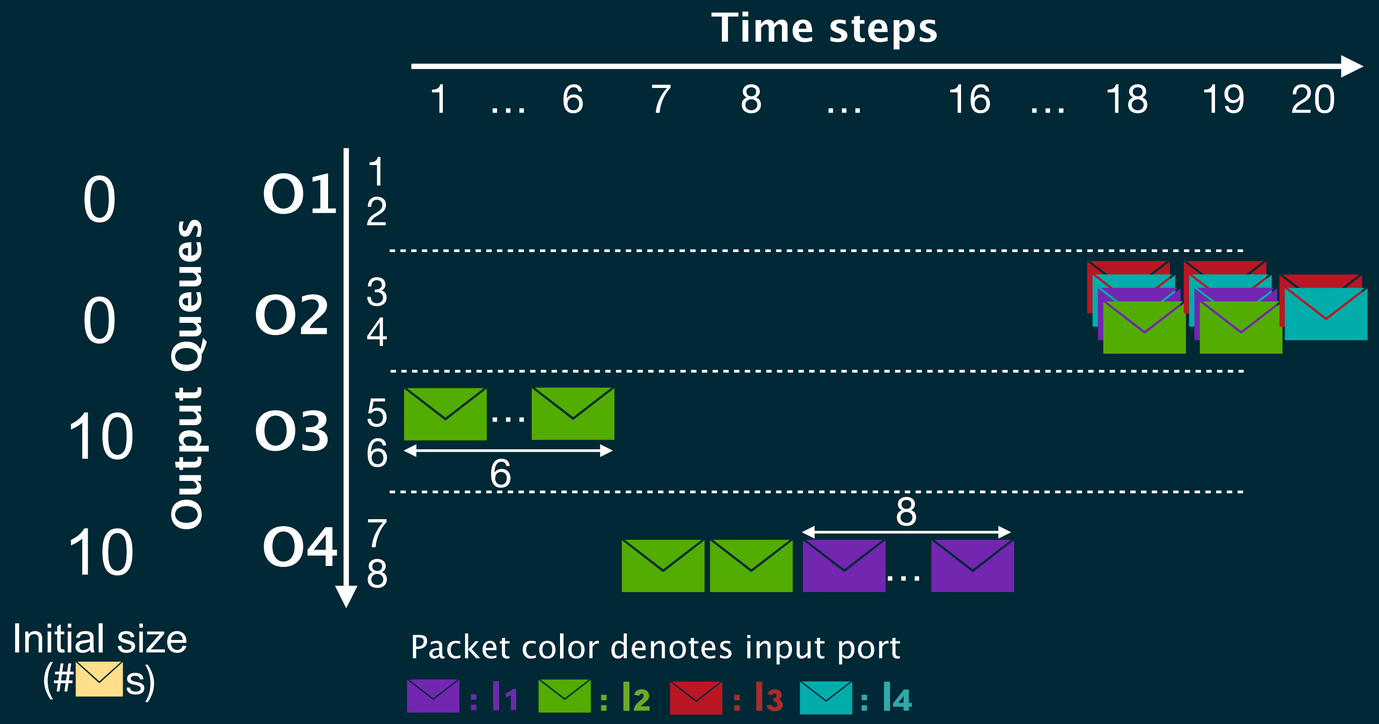


Measurements

Input counts (# s)	Output counts (# s)	Dropped counts (# s)
10	0	
10	3	
3	16	
3	20	
✓	✓	

The packet trace is consistent with input, output counts and the query but not with drop counts

Packet Trace

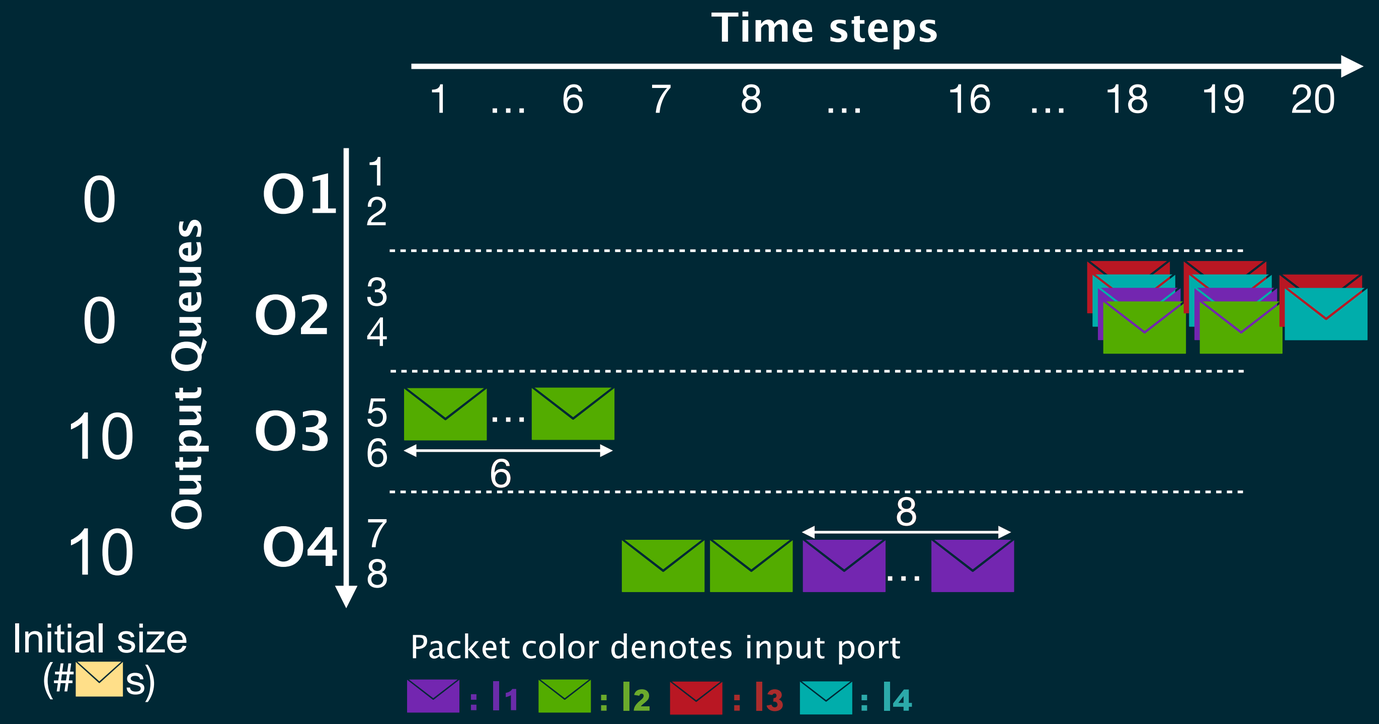


Measurements

Input counts (# s)	Output counts (# s)	Dropped counts (# s)
10	0	0
10	3	0
3	16	0
3	20	2
✓	✓	✗

The packet trace is consistent with input, output counts and the query but not with drop counts, let's start over

Packet Trace

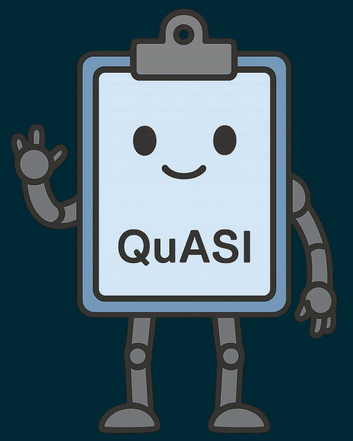


Measurements

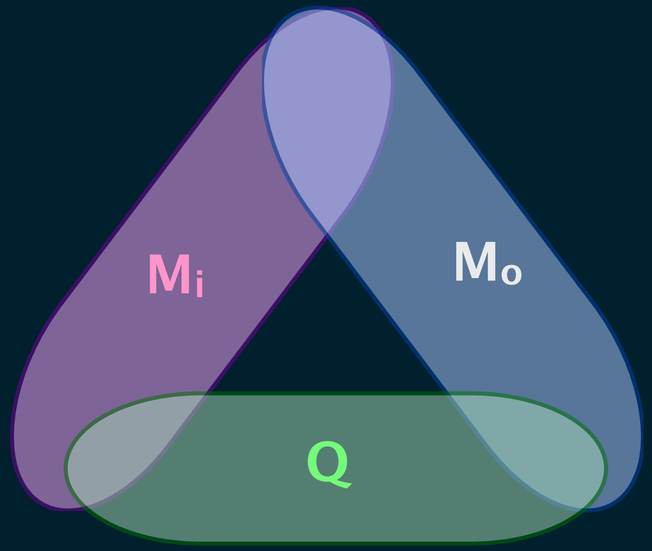
Input counts (# s)	Output counts (# s)	Dropped counts (# s)
10	0	0
10	3	0
3	16	0
3	20	2
✓	✓	✗

There are $\approx 10^{16}$ distinct packet traces that one can generate from the input counts…
proving impossibility requires checking all of them

There are $\approx 10^{16}$ distinct packet traces that one can generate from the input counts…
proving impossibility requires checking all of them

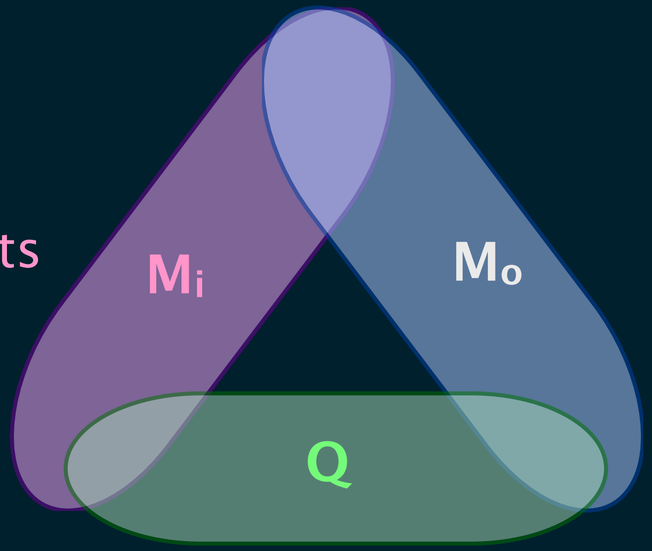


QuASI answers queue-related queries
using per-port incoming, outgoing and dropped packet counts



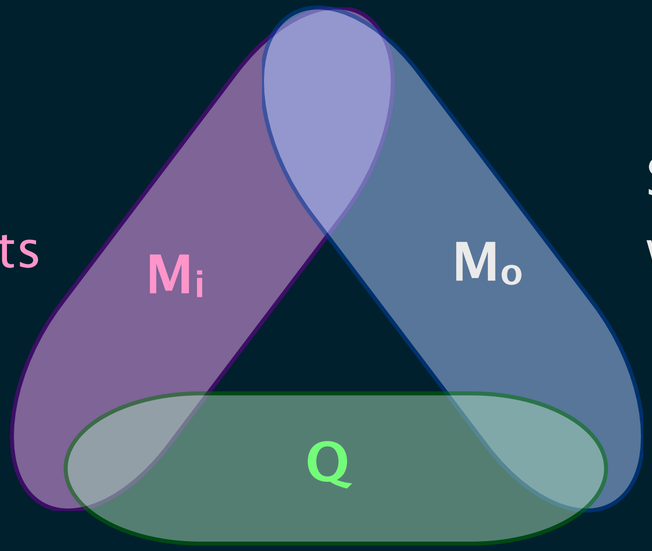
QuASI answers queue-related queries using per-port incoming, outgoing and dropped packet counts

Set of packet traces consistent with input counts



QuASI answers queue-related queries using per-port incoming, outgoing and dropped packet counts

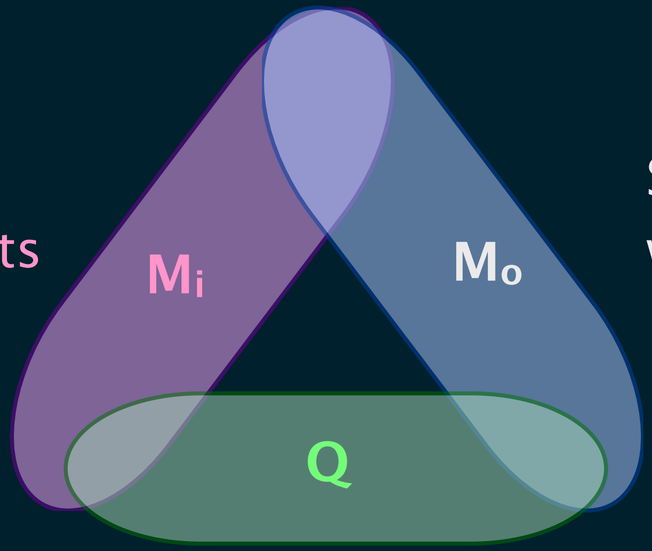
Set of packet traces consistent with input counts



Set of packet traces consistent with output and dropped counts

QuASI answers queue-related queries using per-port incoming, outgoing and dropped packet counts

Set of packet traces consistent with input counts

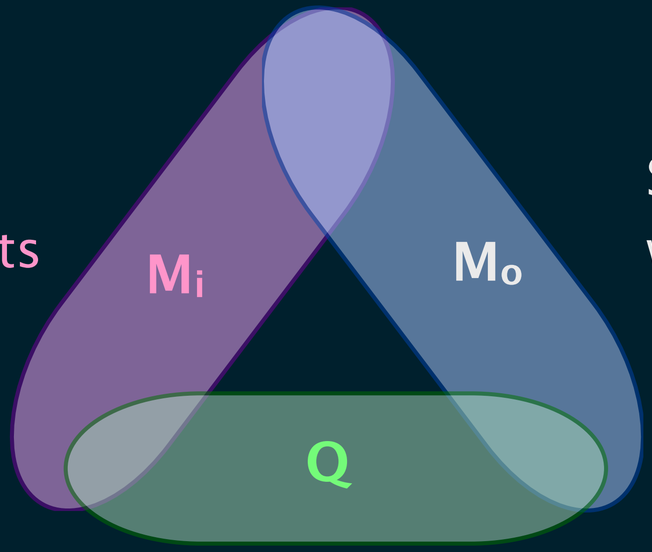


Set of packet traces consistent with output and dropped counts

Set of packet traces consistent with query

QuASI answers queue-related queries
using per-port incoming, outgoing and dropped packet counts

Set of packet traces
consistent with input counts

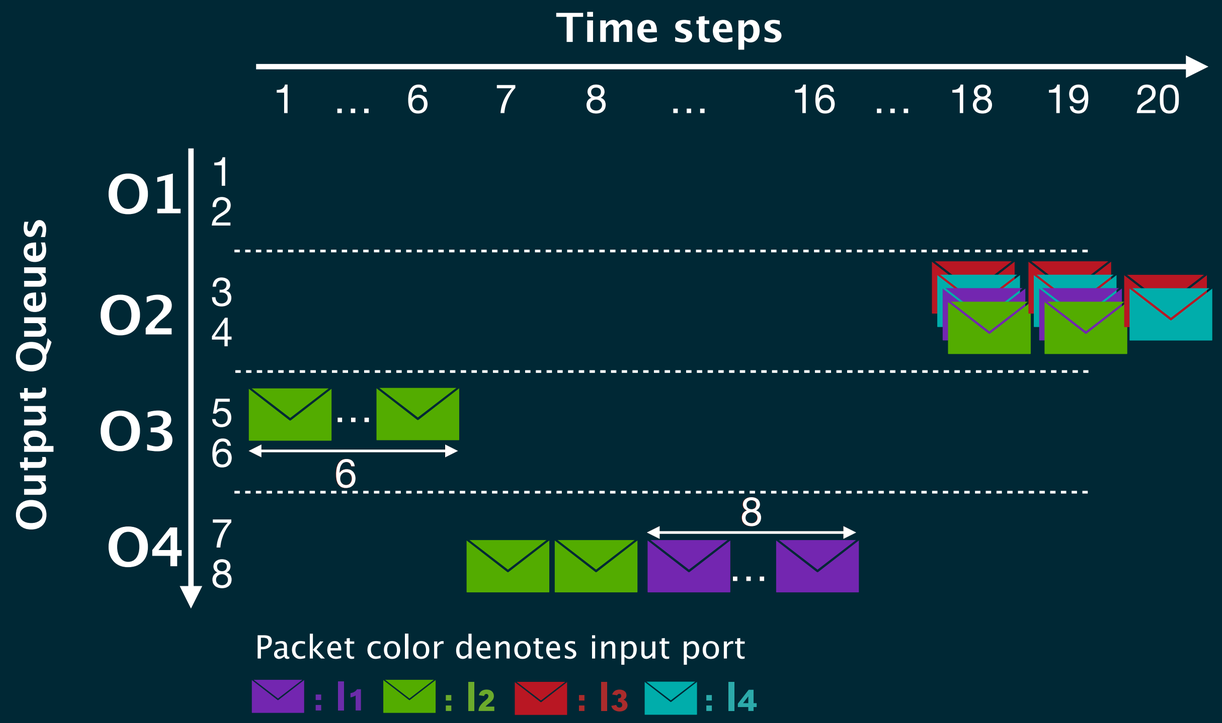


Set of packet traces consistent
with output and dropped counts

Set of packet traces consistent with query

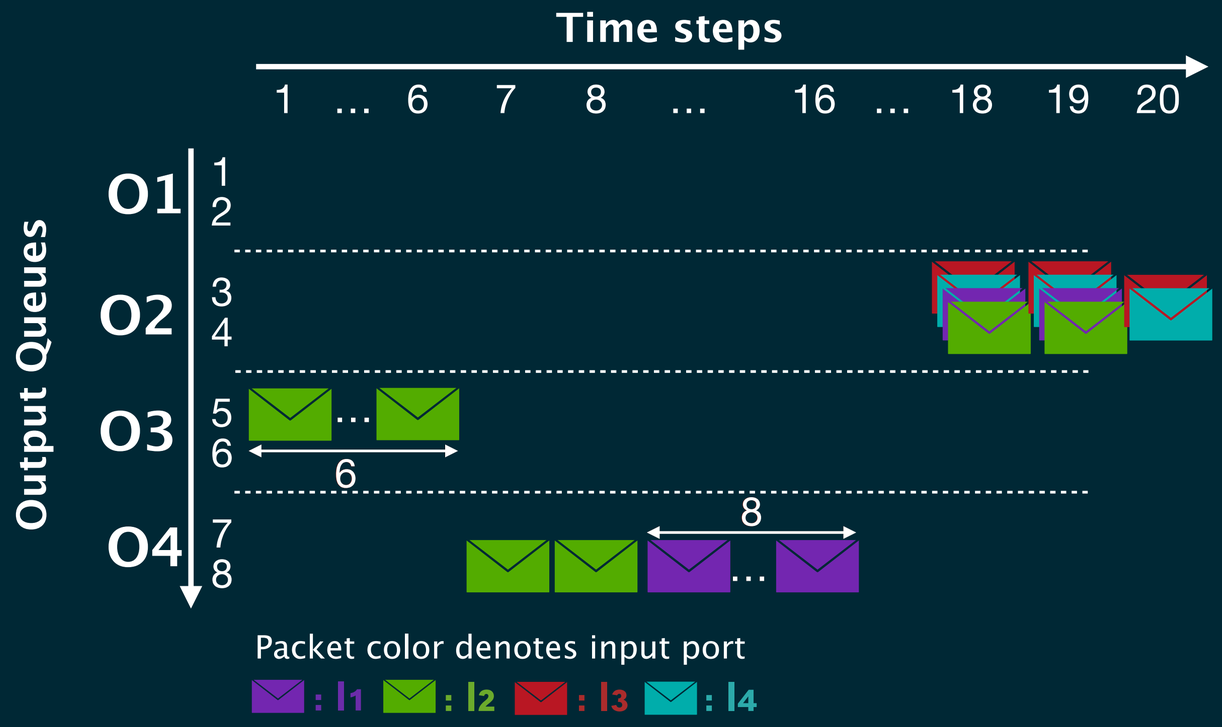
QuASI checks if there is a packet trace in the intersection of those sets

Instead of looking at each packet trace, QuaSi relies on an enqueue abstraction



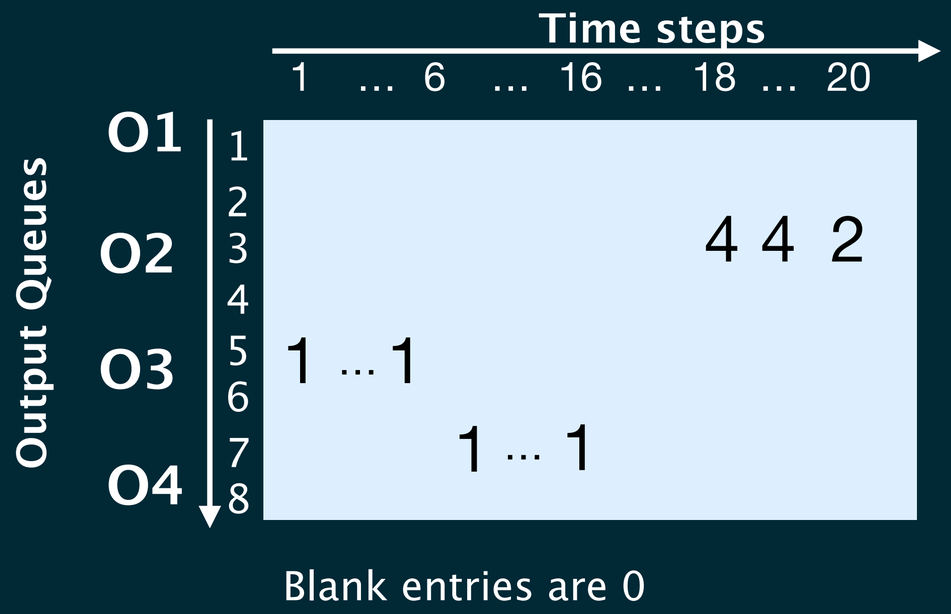
Packet trace

QuaSi reasons about **abstract packet traces:** number of packets that were enqueued per queue and time step



Packet trace

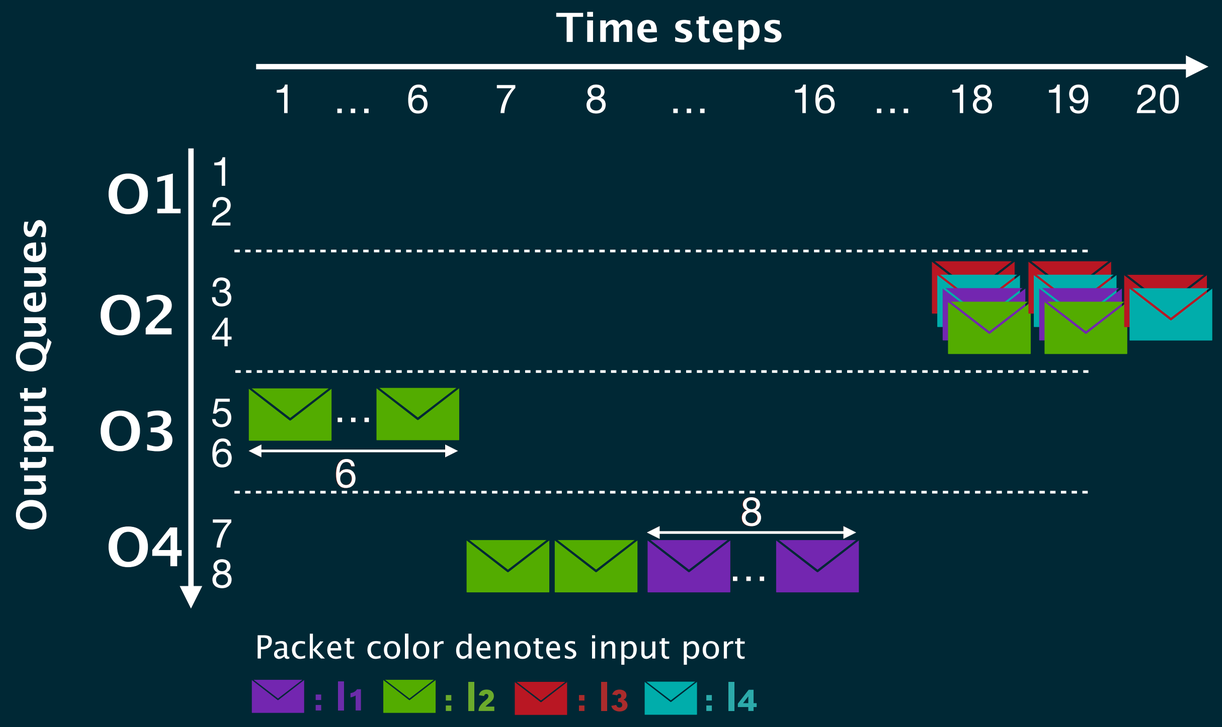
Enqueue-rate Abstraction



Blank entries are 0

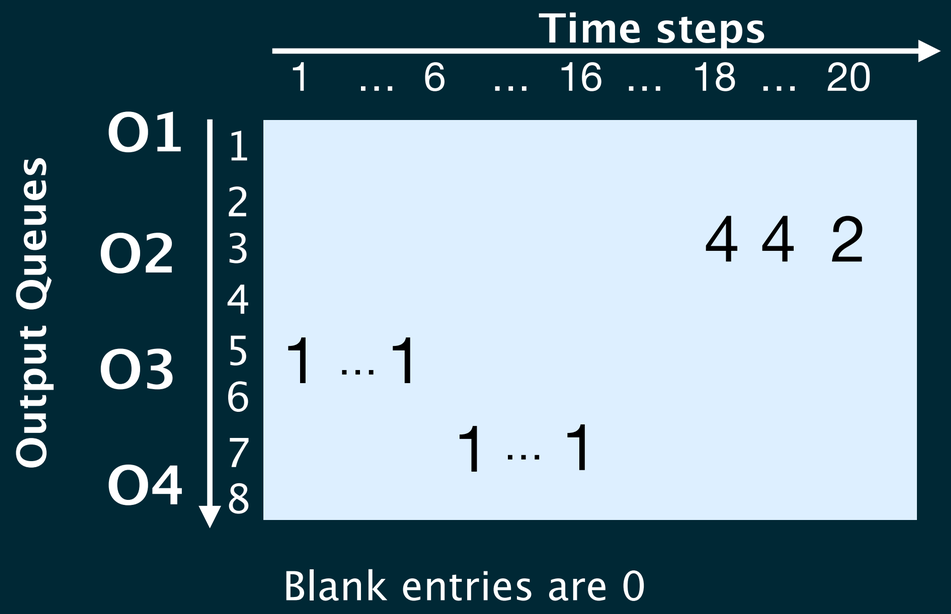
Abstract packet trace

The enqueue abstraction is lossless for the queries we support



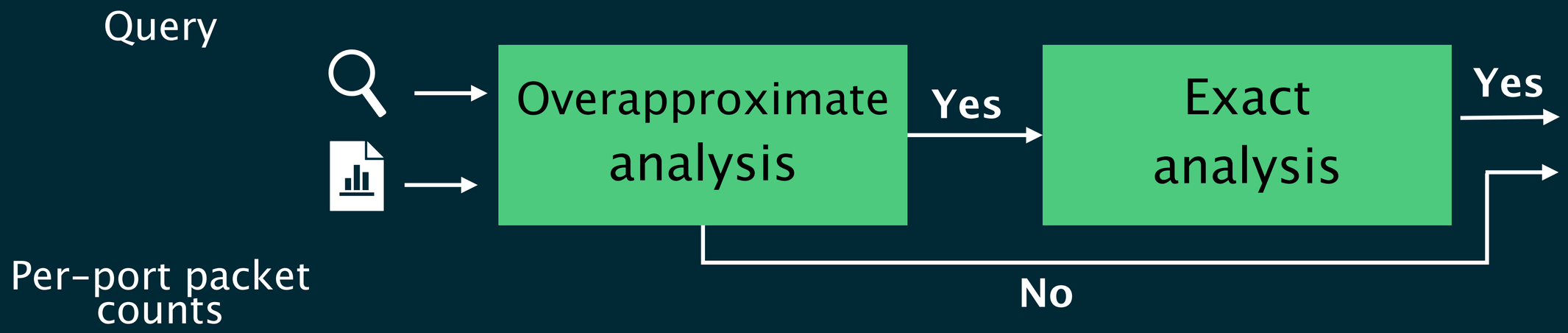
Packet trace

Enqueue-rate Abstraction

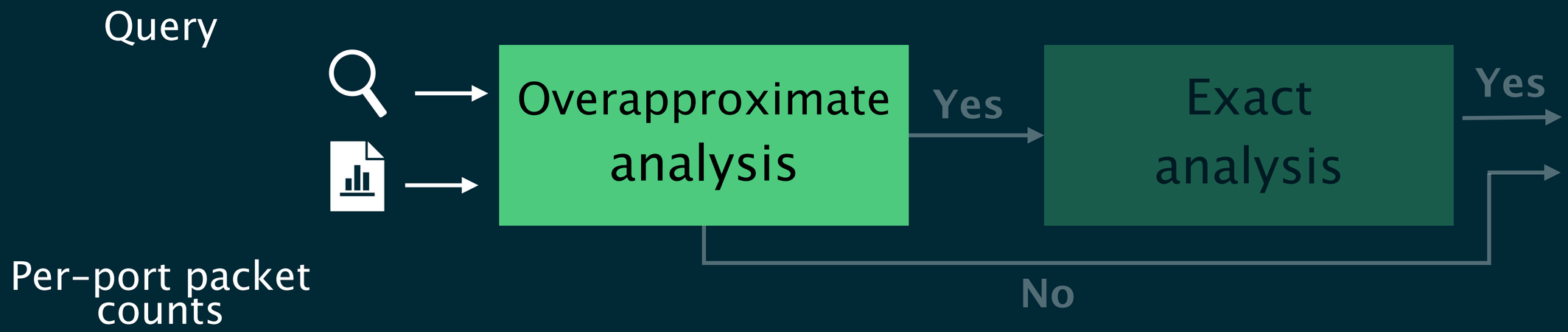


Abstract packet trace

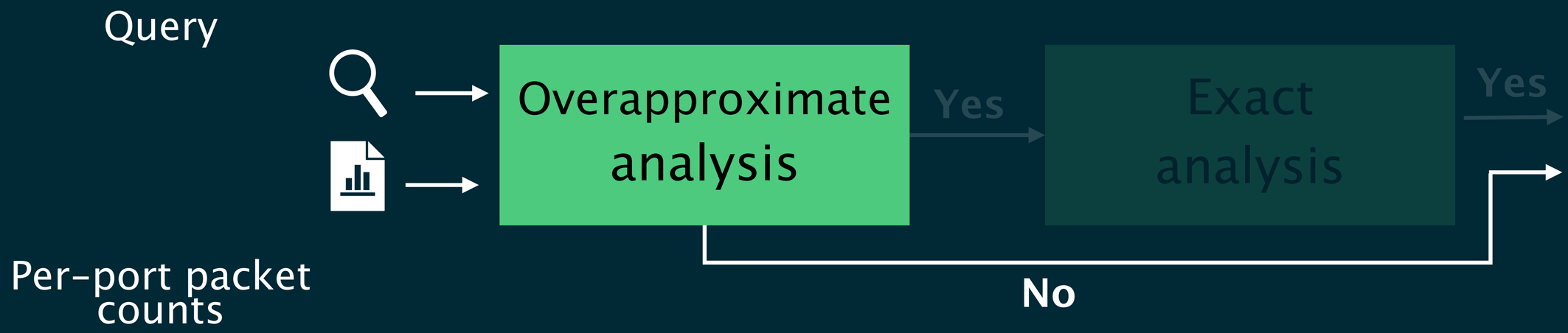
QuASI is inspired by abstraction refinement



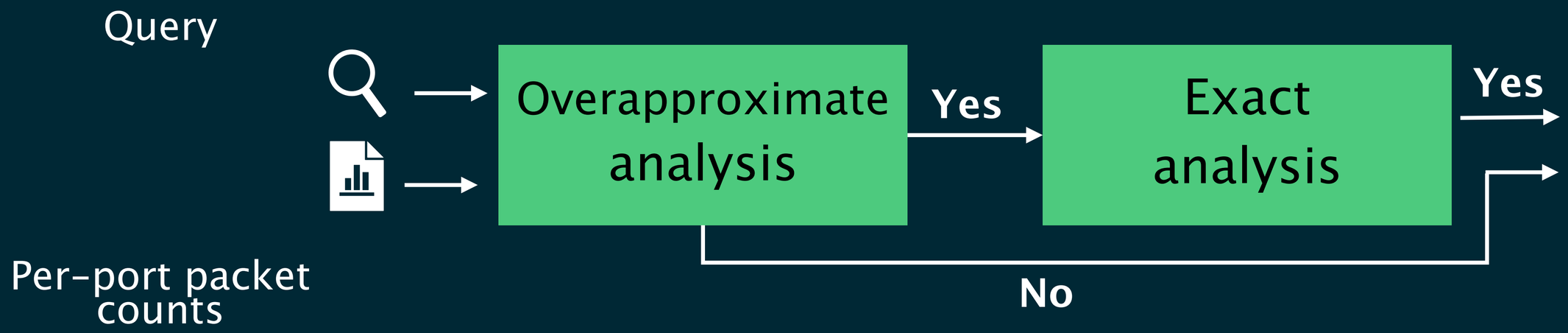
QuASI is inspired by abstraction refinement



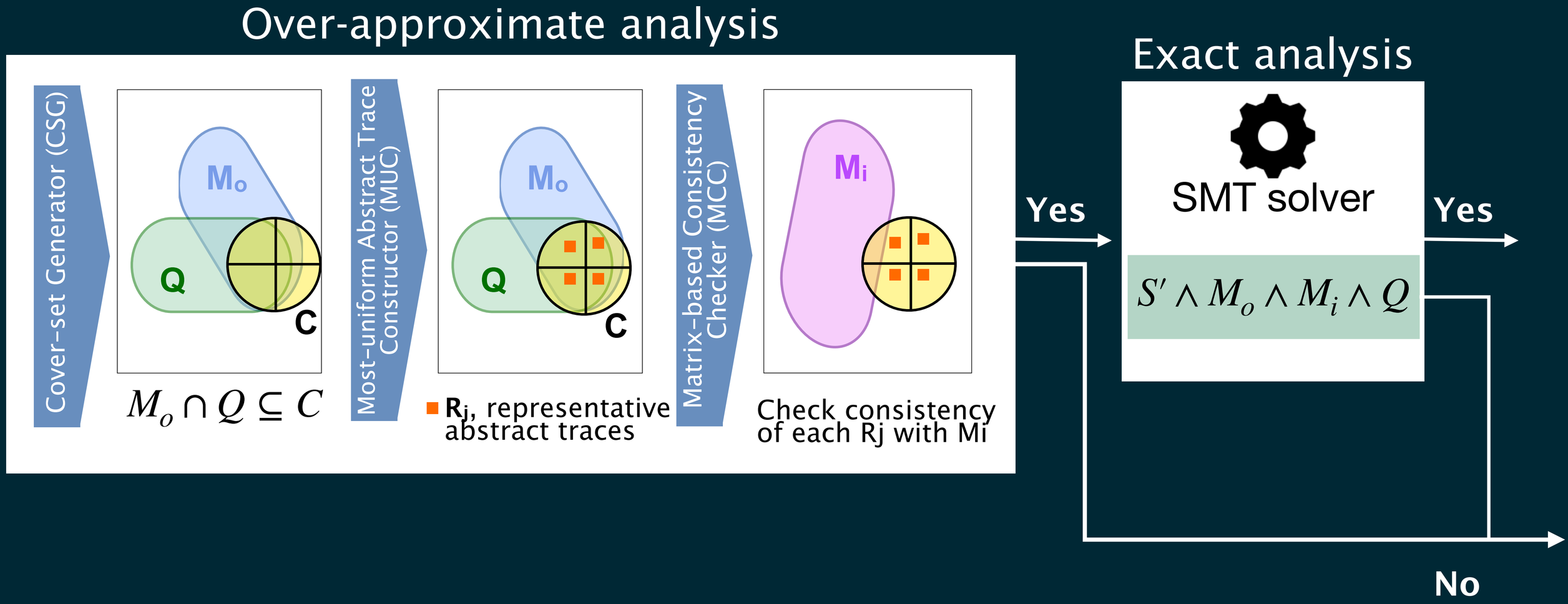
QuASI is inspired by abstraction refinement



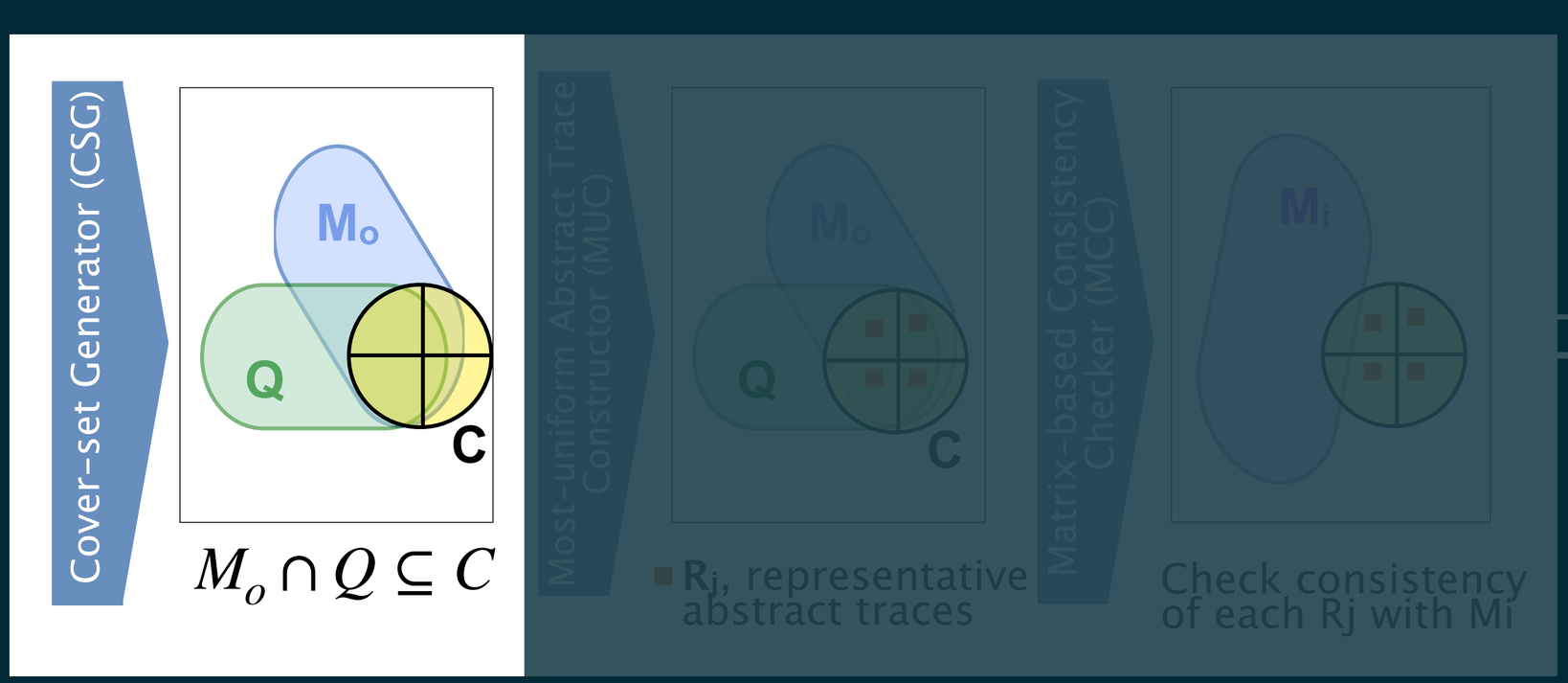
QuASI is inspired by abstraction refinement

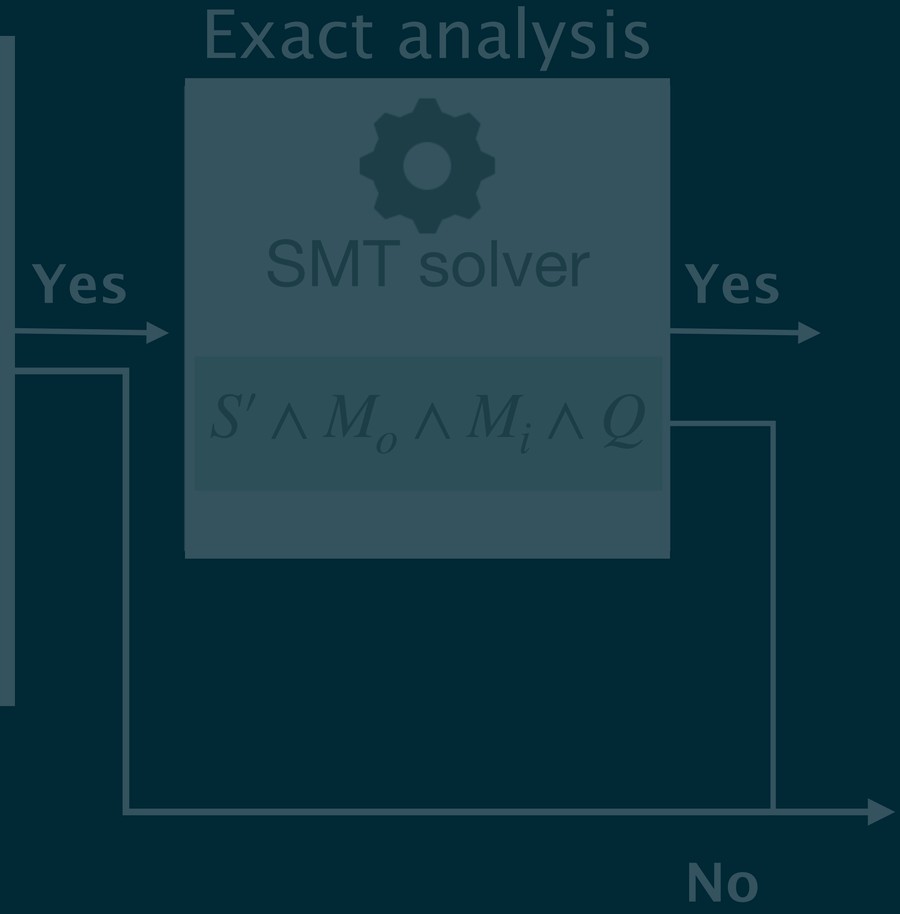


QuASI is inspired by abstraction refinement

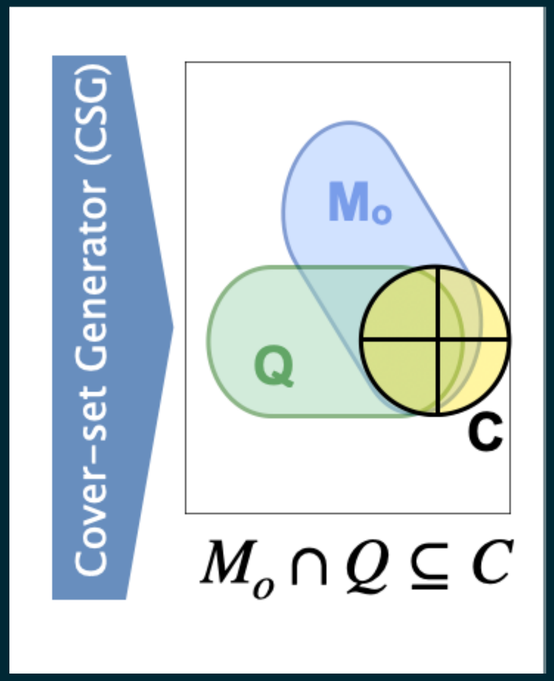


Over-approximate QuASI step 1: The cover-set generator finds necessary conditions for consistency with output counts Mo and the Query Q





Over-approximate QuASI step 1: Cover-set is a disjunction of components, each being a conjunction of expressions over enqueue rate



$$F_C^k$$

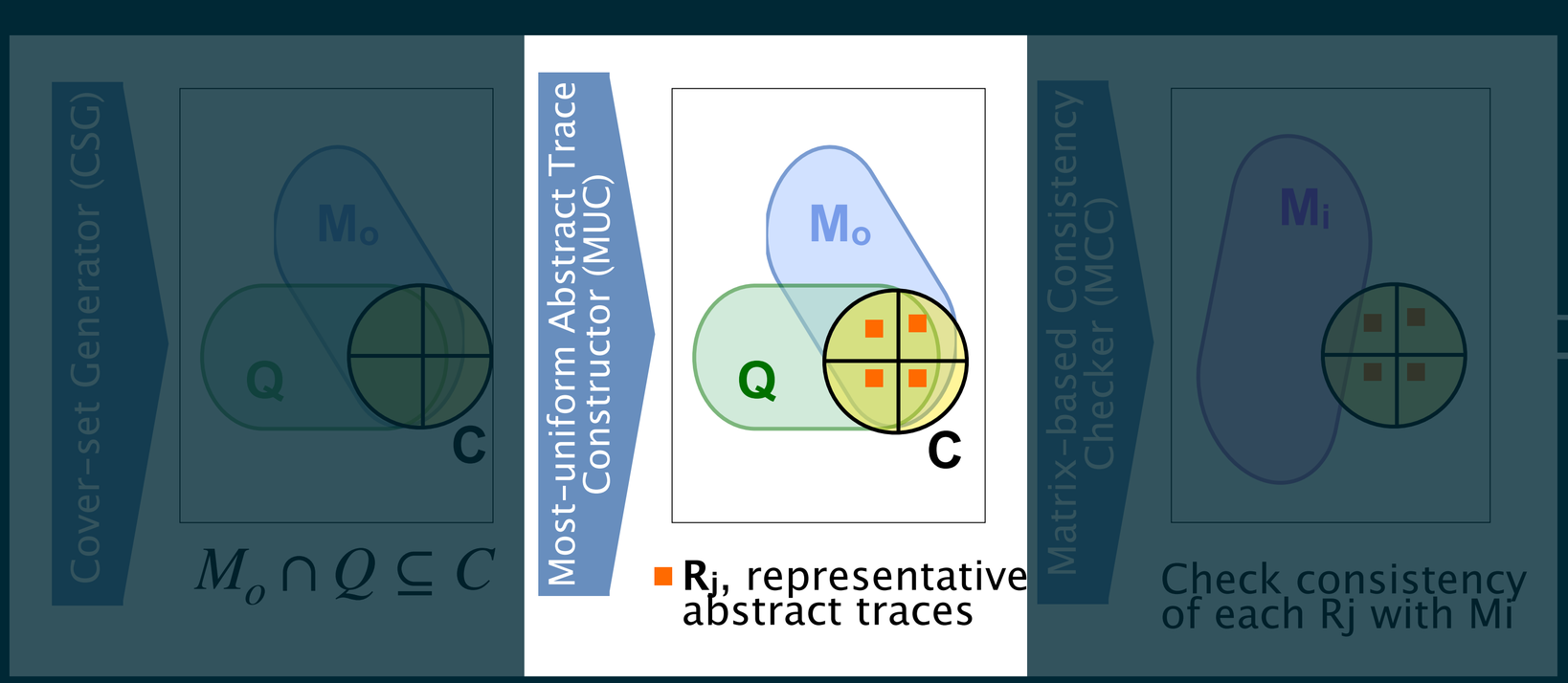
$$\sum_{q \in O_2} cenq(q,T) \geq 10 \ \wedge \sum_{q \in O_3} cenq(q,T) \geq 6 \ \wedge \sum_{q \in O_4} cenq(q,T) \geq 12$$

$$F_C^1$$

$$\sum_{q \in O_2} cenq(q,T) \geq 10 \ \wedge \sum_{q \in O_3} cenq(q,T) \geq 6 \ \wedge \sum_{q \in O_4} cenq(q,T) \geq 12$$

Each component F_C^i is conjunction of expressions over enqueue rate

Over-approximate QuASI step 2: MUC generates a representative abstract trace for each cover-set component



Yes

Exact analysis

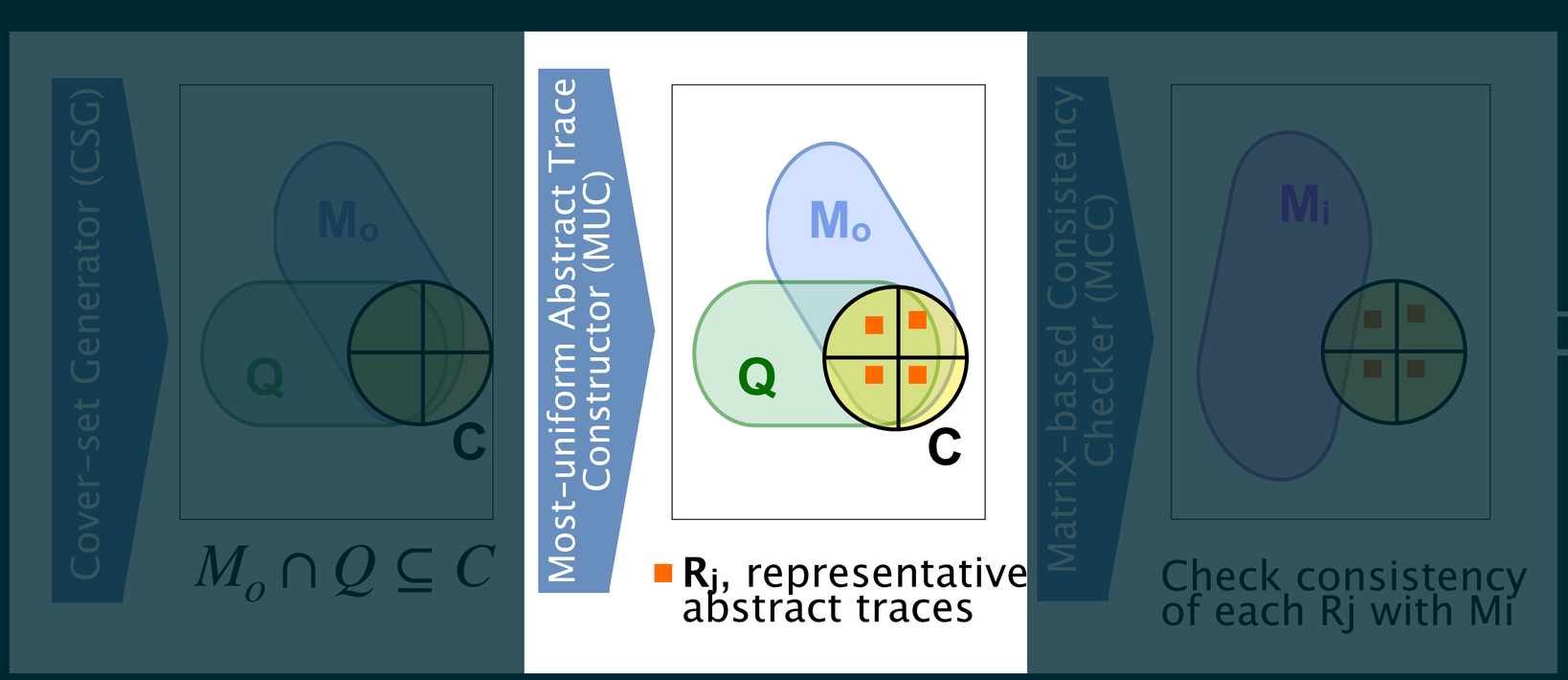
SMT solver

$S' \wedge M_o \wedge M_i \wedge Q$

Yes

No

Over-approximate QuASI step 2: If the representative trace is not consistent with input counts, then no trace in the cover-set component is



Yes

Exact analysis

SMT solver

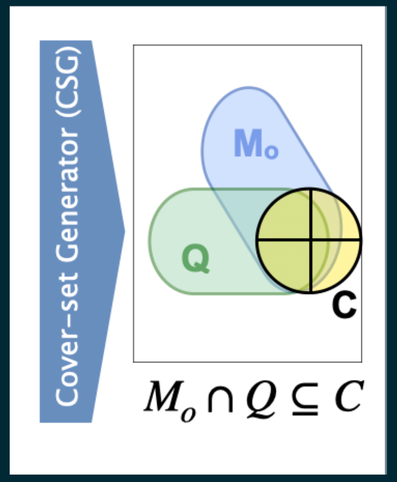
$S' \wedge M_o \wedge M_i \wedge Q$

Yes

No

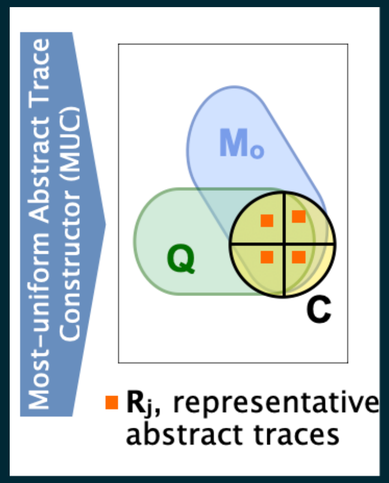
See paper for proof

Over-approximate QuASI step 2: The representative abstract trace is the most uniform allocation subject to the constraints in the component

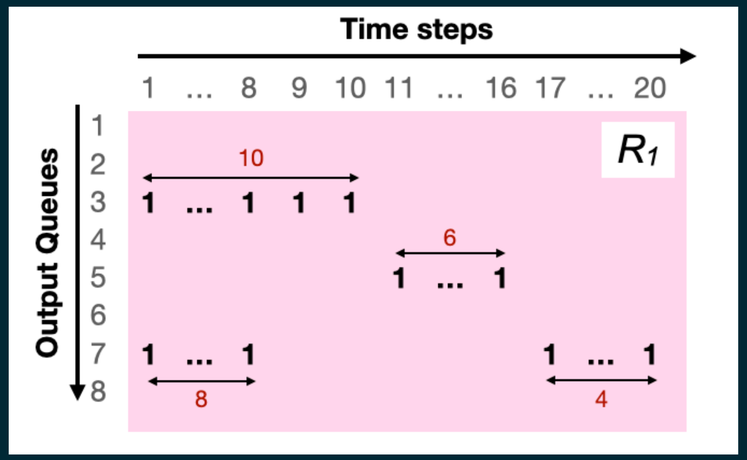


cover-set component F_C^1

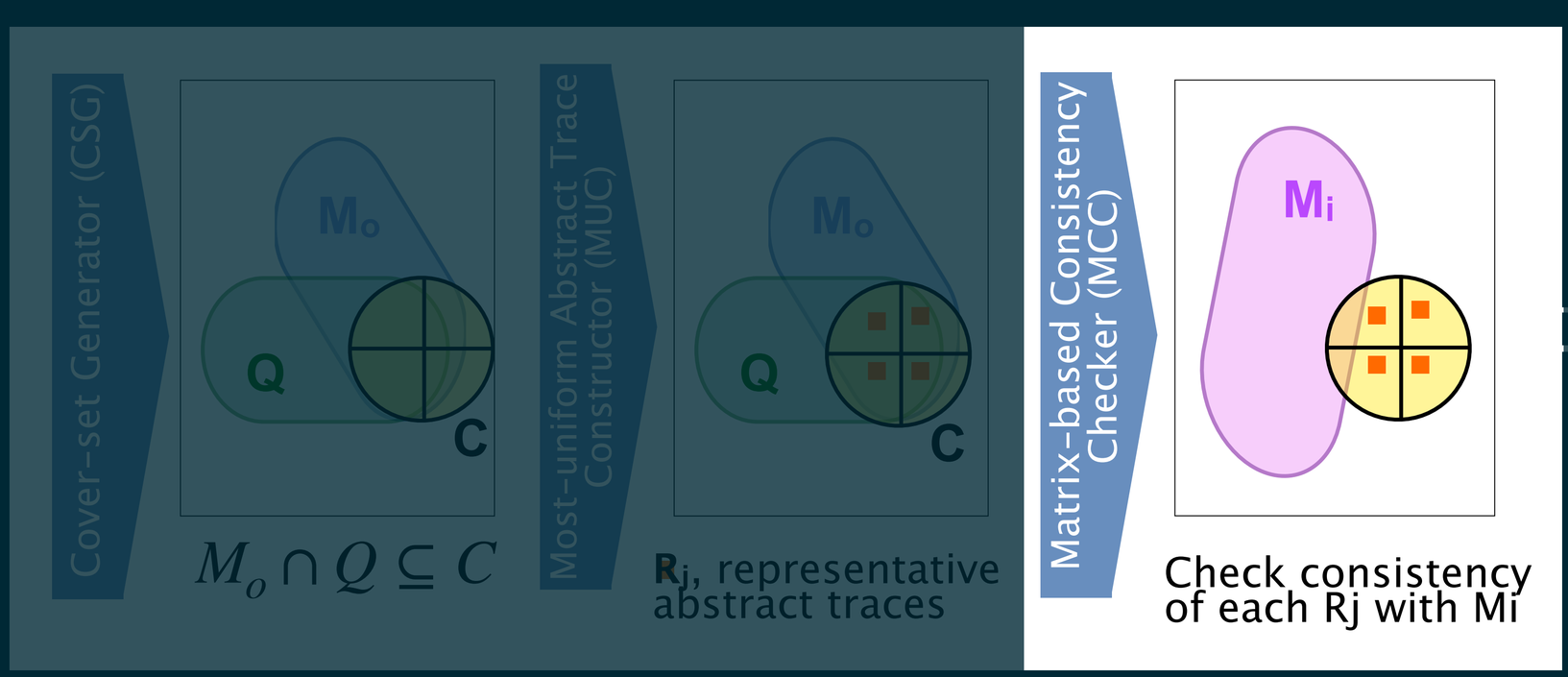
$$\sum_{q \in O_2} cenq(q,T) \geq 10 \ \wedge \sum_{q \in O_3} cenq(q,T) \geq 6 \ \wedge \sum_{q \in O_4} cenq(q,T) \geq 12$$



representative abstract trace R1



Over-approximate QuASI step 3: QuASI checks consistency of each representative abstract trace with M_i using the Gale-Ryser theorem[1,2]



Yes

Exact analysis

SMT solver

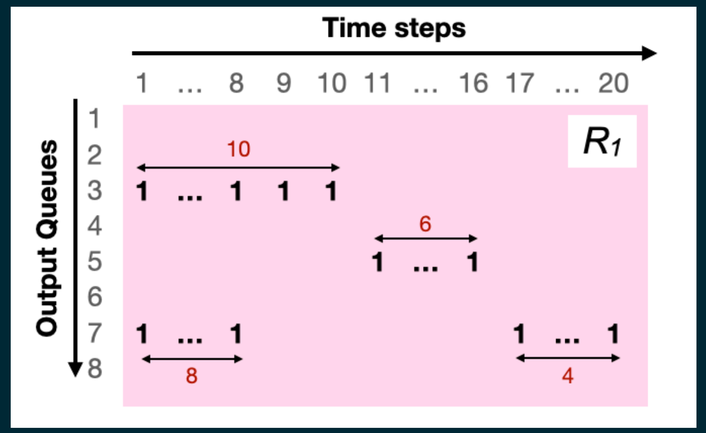
$S' \wedge M_o \wedge M_i \wedge Q$

Yes

No

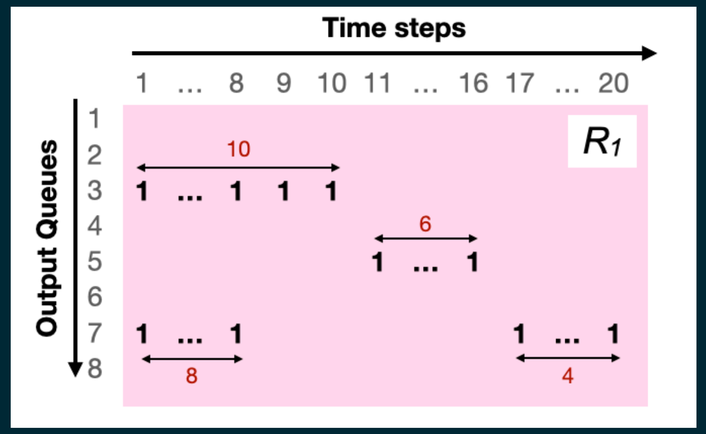
1. David Gale, *Classic Papers in Combinatorics* 1956
2. H. J. Ryser, *Canadian Journal of Mathematics* 1957

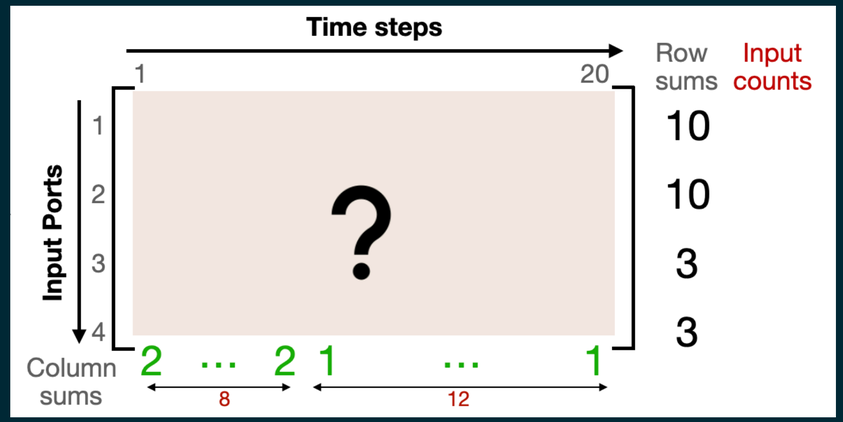
Over-approximate QuASI step 3: QuASI checks consistency of each representative abstract trace with M_i using the Gale-Ryser theorem[1,2]



Is the R1 representative trace consistent with the input counts?

Over-approximate QuASI step 3: QuASI checks consistency of each representative abstract trace with M_i using the Gale-Ryser theorem[1,2]

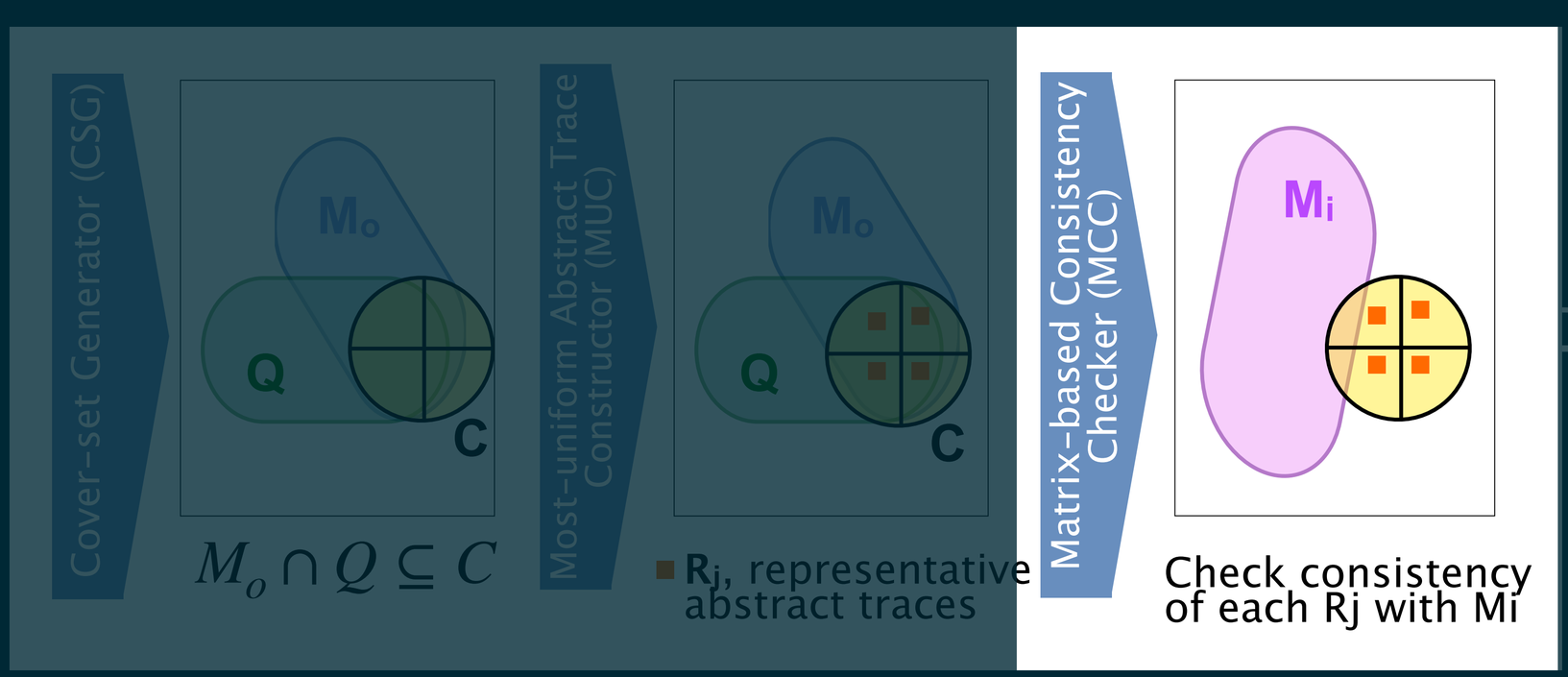




Is the R1 representative trace consistent with the input counts?

Is there a (0,1)-matrix with row sums eq. input packet counts and column sums eq. total number of packets sent /time steps?

Over-approximate QuASI step 3: QuASI checks consistency of each representative abstract trace with M_i using the Gale-Ryser theorem[1,2]



Yes

Exact analysis

SMT solver

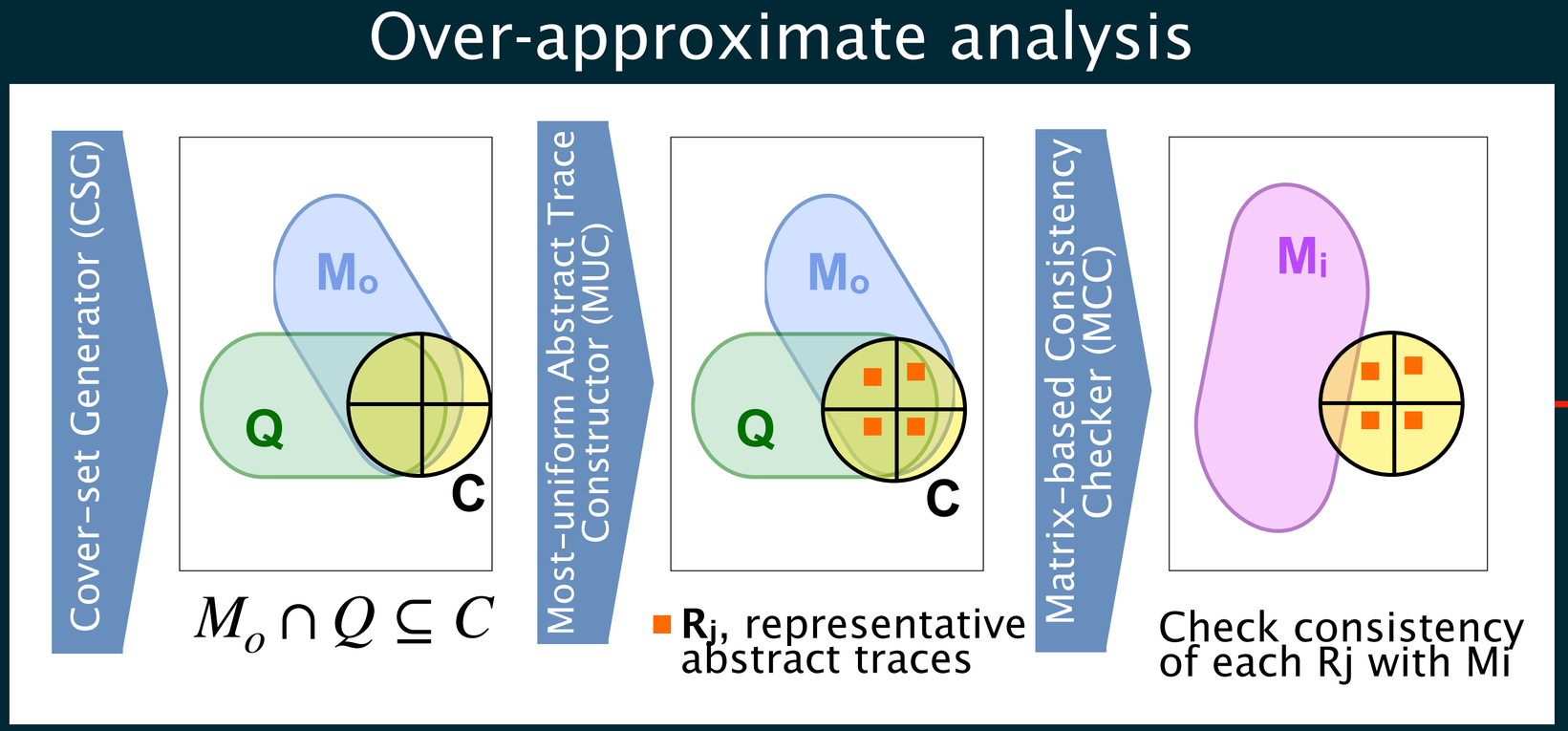
$S' \wedge M_o \wedge M_i \wedge Q$

Yes

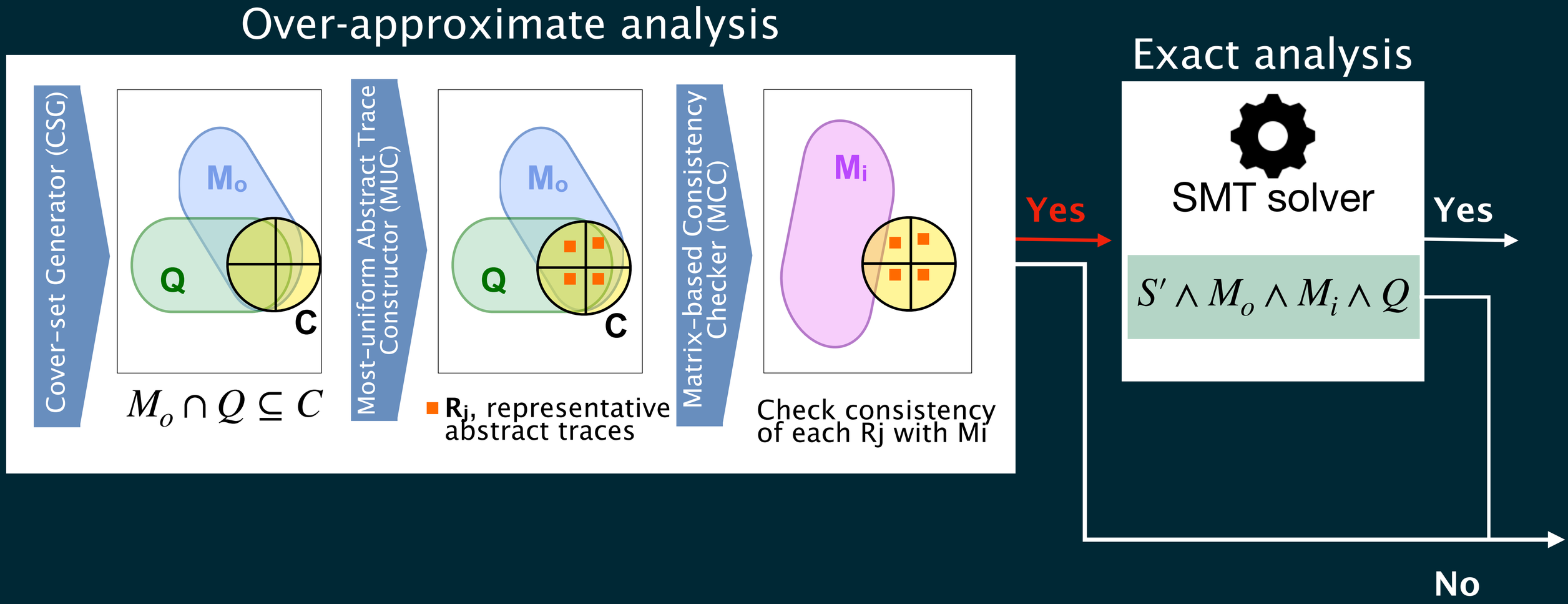
No

1. David Gale, *Classic Papers in Combinatorics* 1956
2. H. J. Ryser, *Canadian Journal of Mathematics* 1957

If Gale-Ryser theorem says none of the representative traces is consistent with the input counts Mi, QuASI concludes— we are done!



Else, QuASI uses an SMT solver to model the exact conditions for consistency with M_i, M_o, and Q



We evaluate QuASI in multiple queue-related queries

BurstOccurrence: Could a burst of rate R and duration D occur?

QlenK: Could the queue length at port Oi be at least K?

MaxQlen: What is the maximum queue length at port I?

MaxBuff: What is the maximum buffer occupancy during the interval?

QuASI evaluation

Is QuASI useful?

Is QuASI better than SOTA?

Is QuASI scalable?

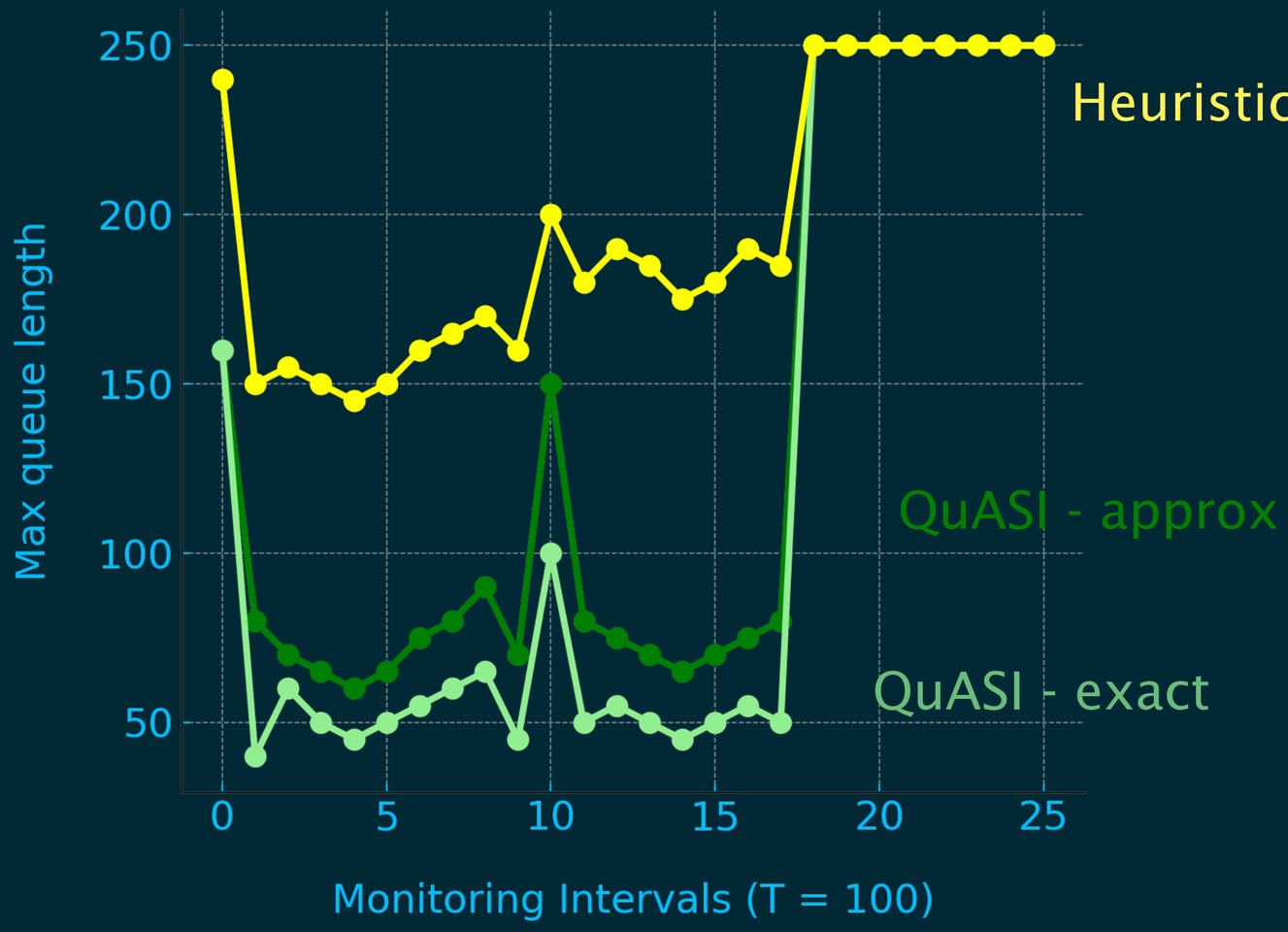
QuASI evaluation

Is QuASI useful?

Is QuASI better than SOTA?

Is QuASI scalable?

QuASI-approx finds bounds up to 58% tighter than the heuristic baseline within 1 second



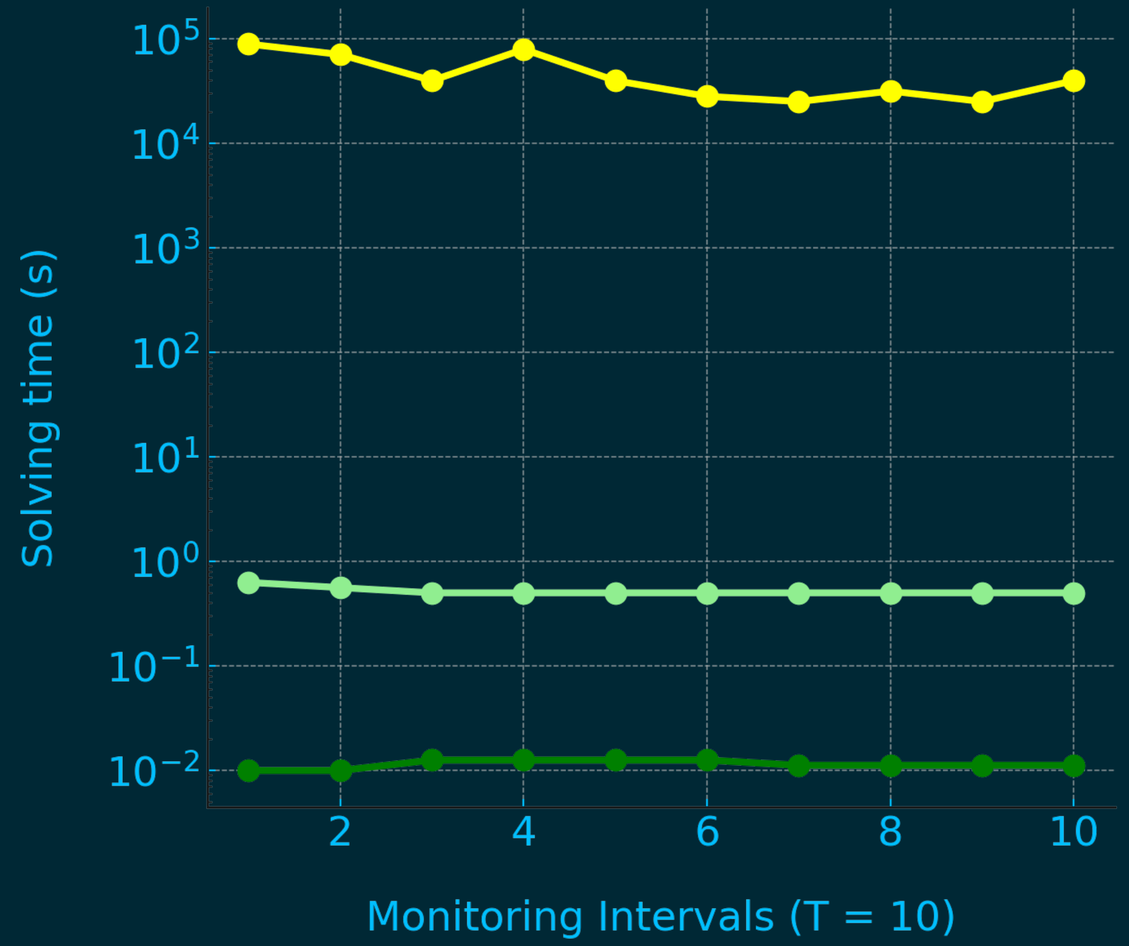
QuASI evaluation

Is QuASI useful?

Is QuASI better than SOTA?

Is QuASI scalable?

QuASI computes the maximum queue length 10^6X faster than FPerf [1]



Heuristic

QuASI - exact

QuASI - approx

[1] Arashloo et al. NSDI 2023

QuASI evaluation

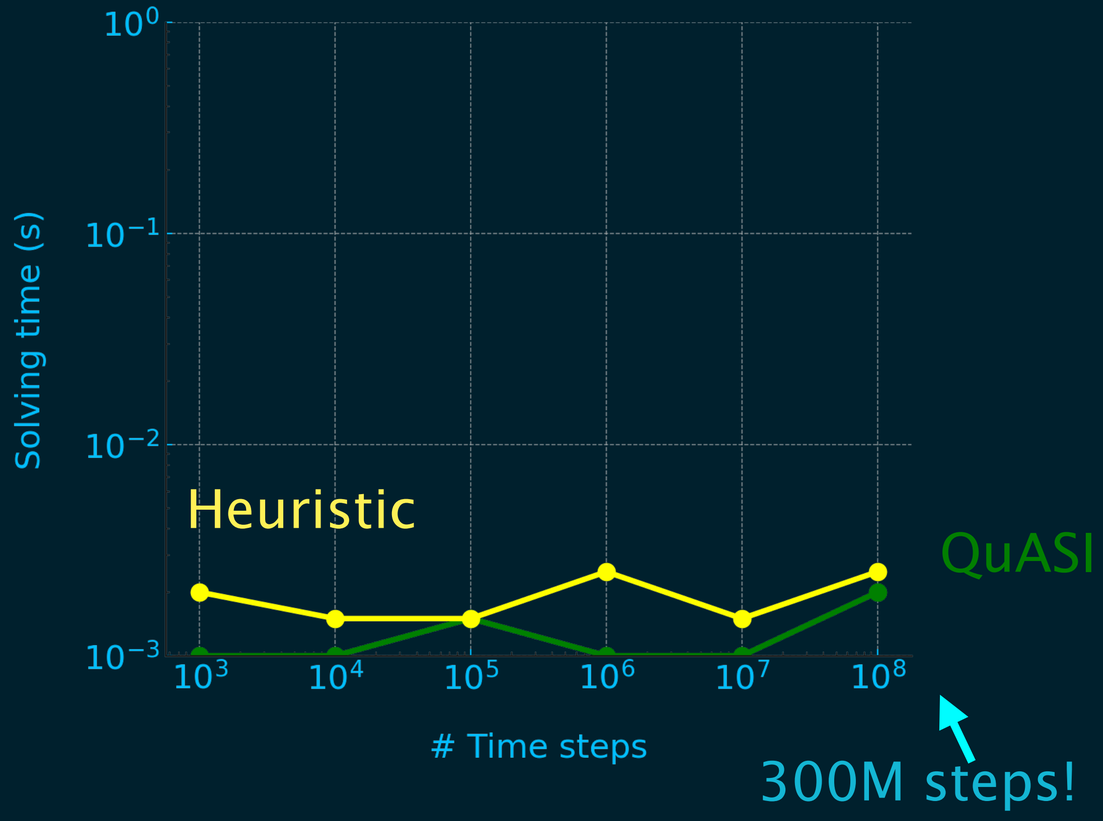
Is QuASI useful?

Is QuASI better than SOTA?

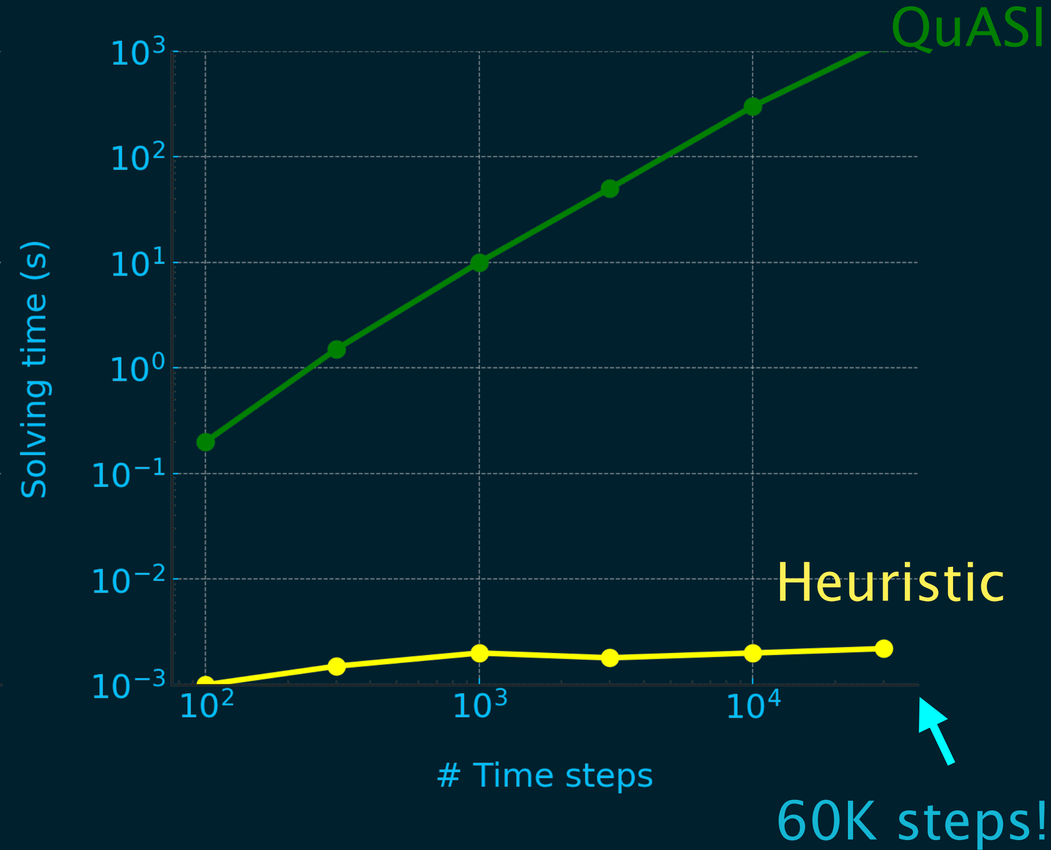
Is QuASI scalable?

QUASI takes almost constant time for QlenK and scales quadratically with interval size for BurstOccurrence.

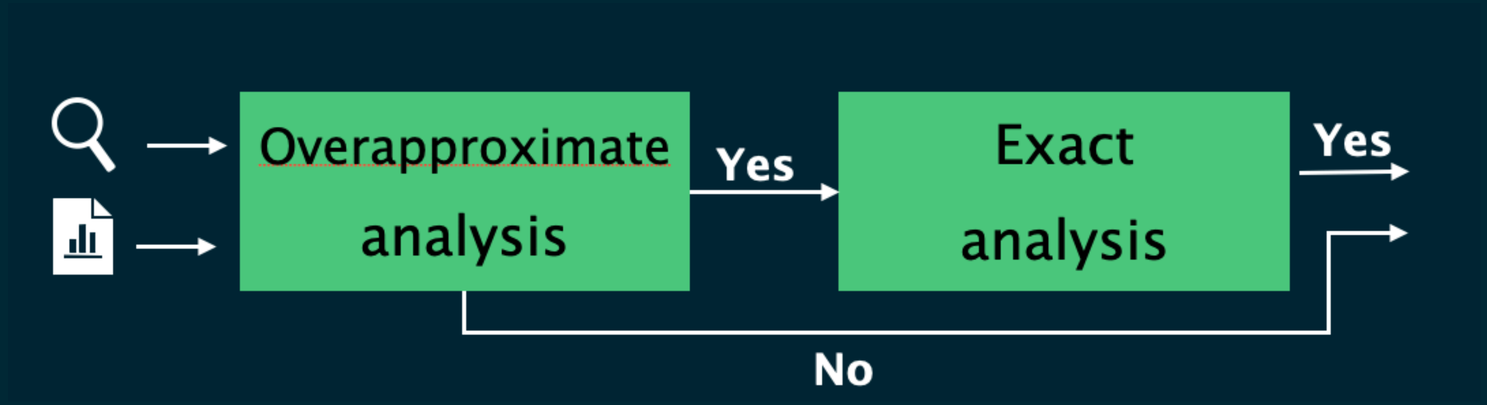
QlenK



Burst Occurrence



A Layered Formal Methods Approach to Answering Queue-related Queries



- QuASI answers queue-related (e.g., Burst occurrence) queries using coarse-grained per-port packet counts
- QuASI uses the enqueue rate abstraction which is lossless for the queries it supports
- QuASI is six orders of magnitude faster than SOTA while giving non-trivial answers

netsyn.princeton.edu

A Layered Formal Methods Approach to Answering Queue-related Queries

Divya Raghunathan, Maria Apostolaki, Aarti Gupta