Example-Guided Synthesis of Relational Queries

Aalok Thakkar

Example-Guided Synthesis of Relational Queries

Aalok Thakkar

Our lab is offering PhD, visiting researcher, and internship positions.
Contact me at aalok.thakkar@ashoka.edu.in for more details.

Example-Guided Synthesis of Relational Queries

Example-Guided Synthesis of Relational Queries

Example-Guided Synthesis of Relational Queries

declarative logic programs

SQL

Datalog

Cypher

SPARQL

Example-Guided Synthesis of Relational Queries

declarative logic programs

PQL

Prolog

LogiQL

CodeQL

$$q = R \mid \sigma_c(q) \mid \pi_\alpha(q) \mid q \bowtie_\theta q$$

$$q = R \mid \sigma_c(q) \mid \pi_\alpha(q) \mid q \bowtie_\theta q$$

$$q = R \mid \pi_\alpha(q) \mid q \bowtie_\theta q$$

$$q = R \mid \sigma_c(q) \mid \pi_\alpha(q) \mid q \bowtie_\theta q$$

$$q = R \mid \pi_\alpha(q) \mid q \bowtie_\theta q$$

PLDI 2021: $q = R \mid \pi_\alpha(q) \mid q \bowtie_\theta q \mid q \cup q$

VLDB 2023: $q = R \mid \sigma_c(q) \mid \pi_\alpha(q) \mid q \bowtie_\theta q \mid q \cup q$

OOPSLA 2023: $q = R \mid \sigma_c(q) \mid \pi_\alpha(q) \mid q \bowtie_\theta q \mid q \cup q \mid$ recursion

$$q = R \mid \sigma_c(q) \mid \pi_\alpha(q) \mid q \bowtie_\theta q$$

$$q = R \mid \pi_\alpha(q) \mid q \bowtie_\theta q$$

PLDI 2021: $q = R \mid \pi_\alpha(q) \mid q \bowtie_\theta q \mid q \cup q$

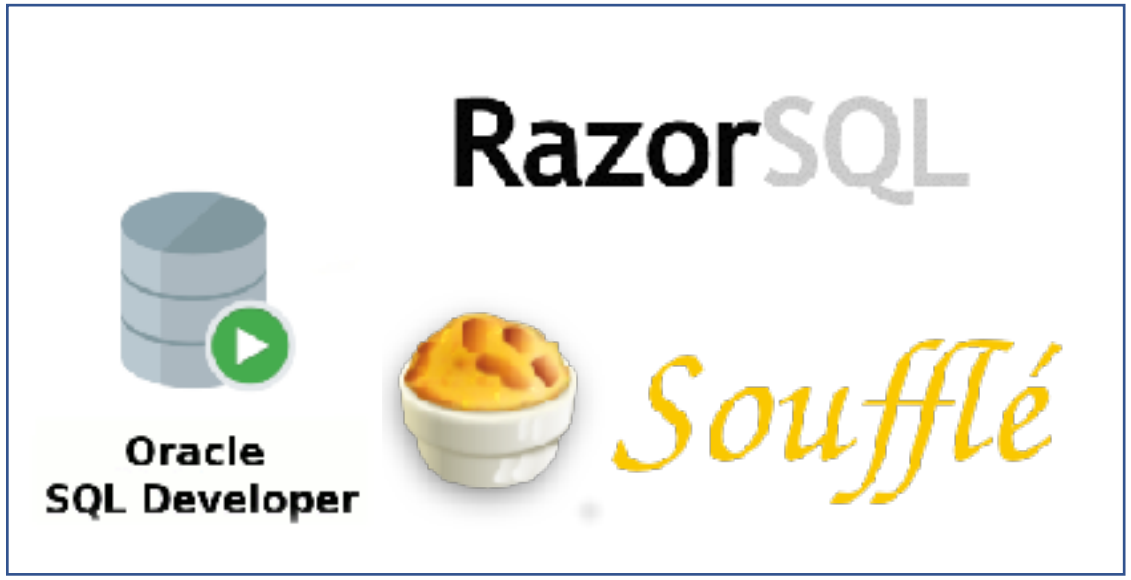
VLDB 2023: $q = R \mid \sigma_c(q) \mid \pi_\alpha(q) \mid q \bowtie_\theta q \mid q \cup q$

OOPSLA 2023: $q = R \mid \sigma_c(q) \mid \pi_\alpha(q) \mid q \bowtie_\theta q \mid q \cup q \mid$ recursion

Program Analysis

Network Analysis

Knowledge Discovery

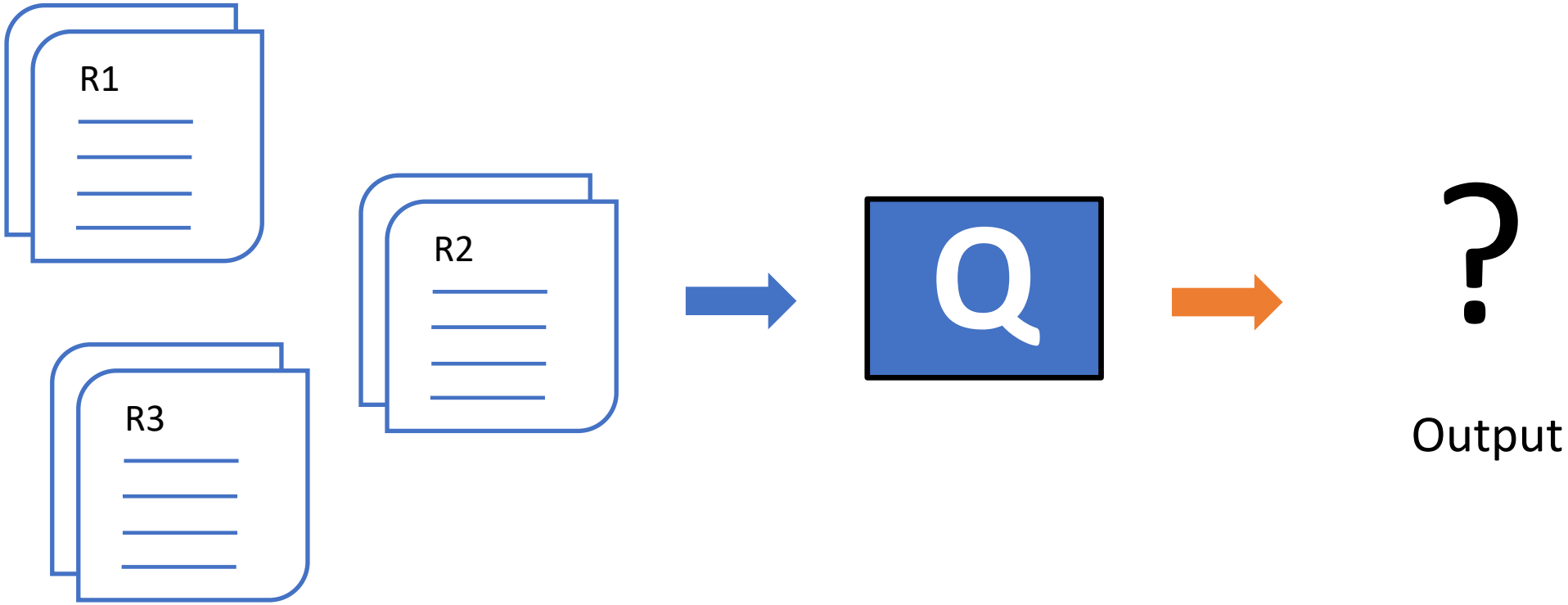


Database Querying

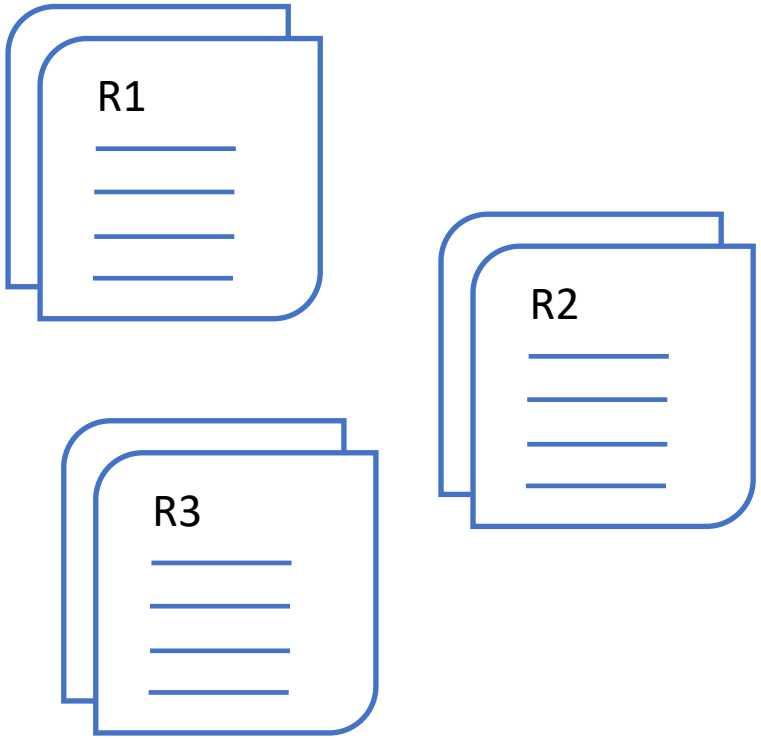
Industrial Applications

Example-Guided Synthesis of Relational Queries

Querying



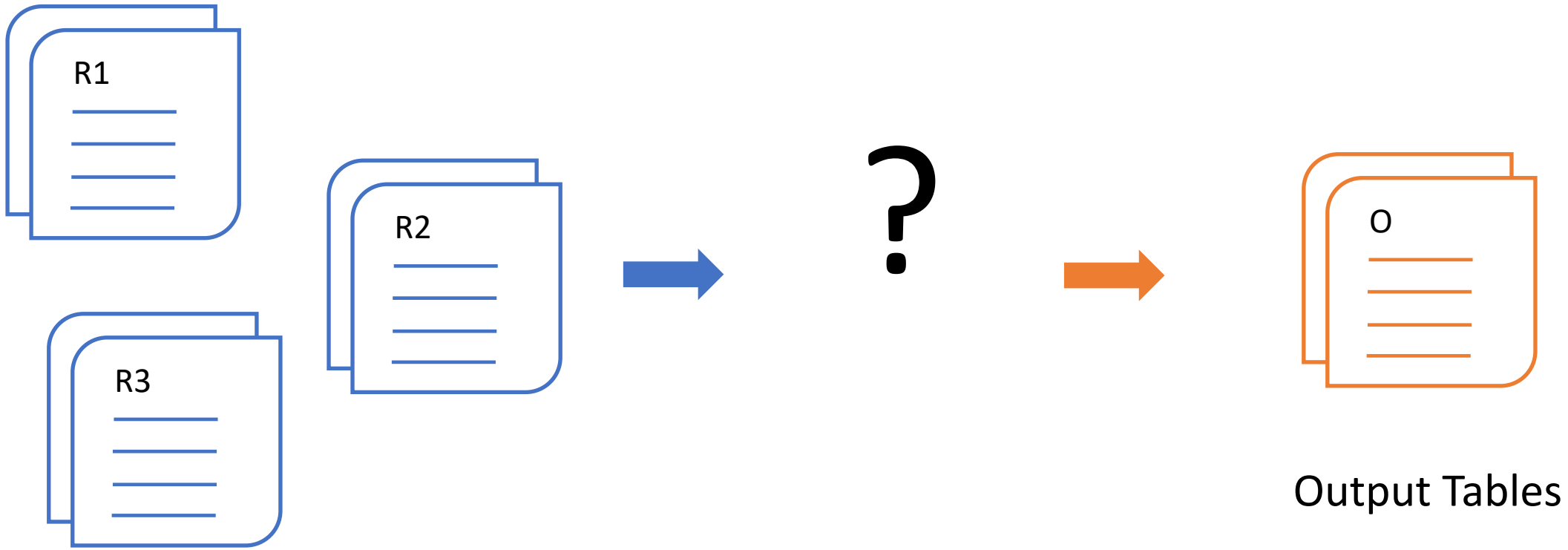
Query Synthesis Problem



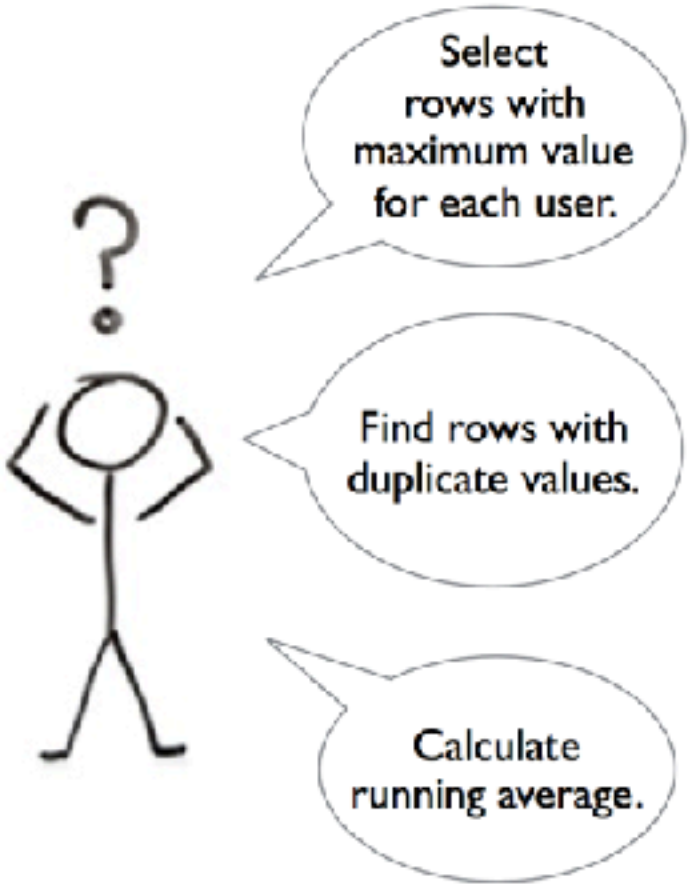
Input Tables

Output Tables

Query Synthesis Problem



End User



stackoverflow

SQL

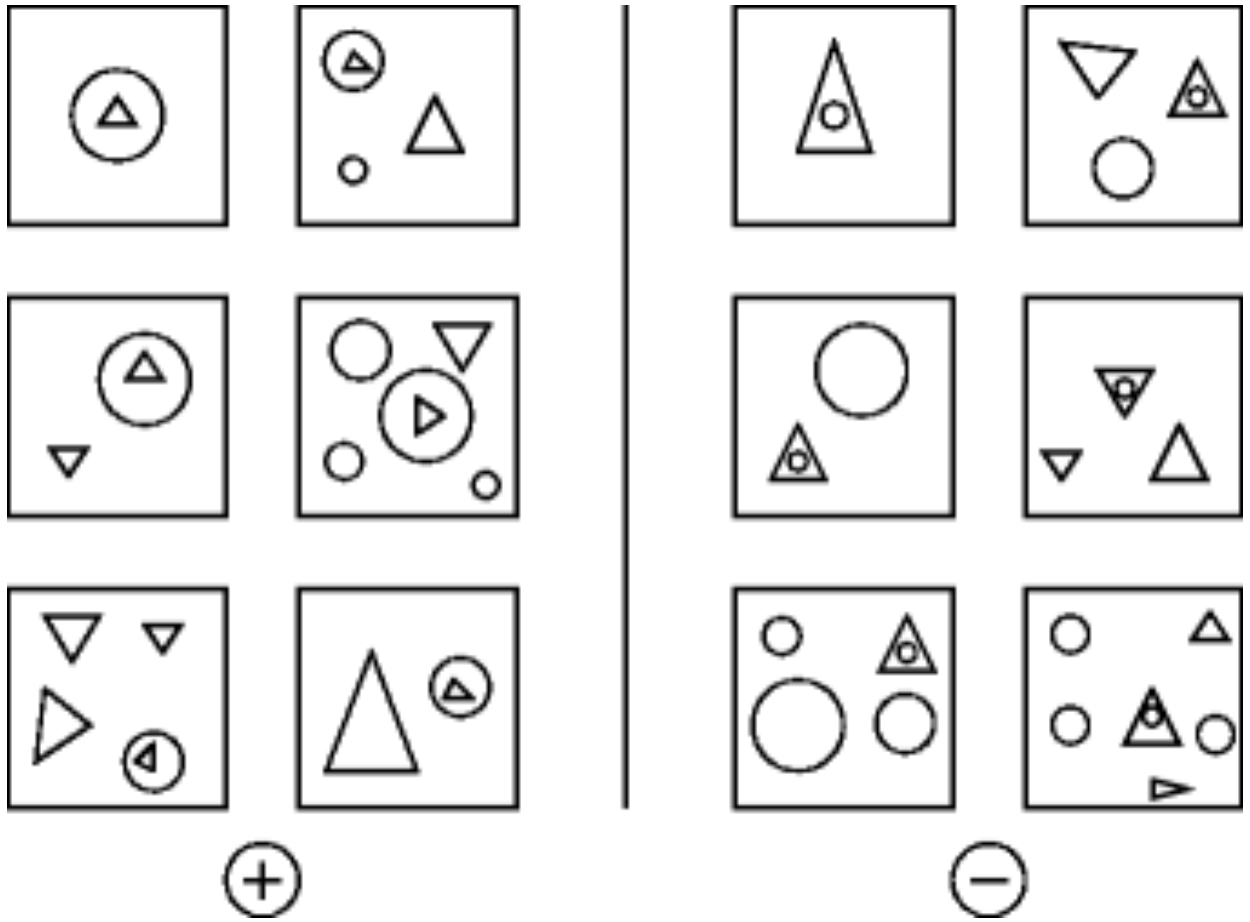
```sql
Select x.id, x.customer, x.total
From PURCHASES x
Join (Select p.customer,
             Max(total)
      From PURCHASES p
      Group By p.customer) y
On y.customer = x.customer
   And y.max_total = x.total
```

```sql
Select *
From  Users a
Where Exists
   (Select *
    From Users b
    Where (a.name = b.name
           Or  a.email = b.email)
          And a.ID <> b.id)
```

```sql
Select a.ord, a.val, Avg(b.val)
From t As a Join t As b
Where b.ord <= a.ord
Group By a.ord,a.val
Order By a.ord
```

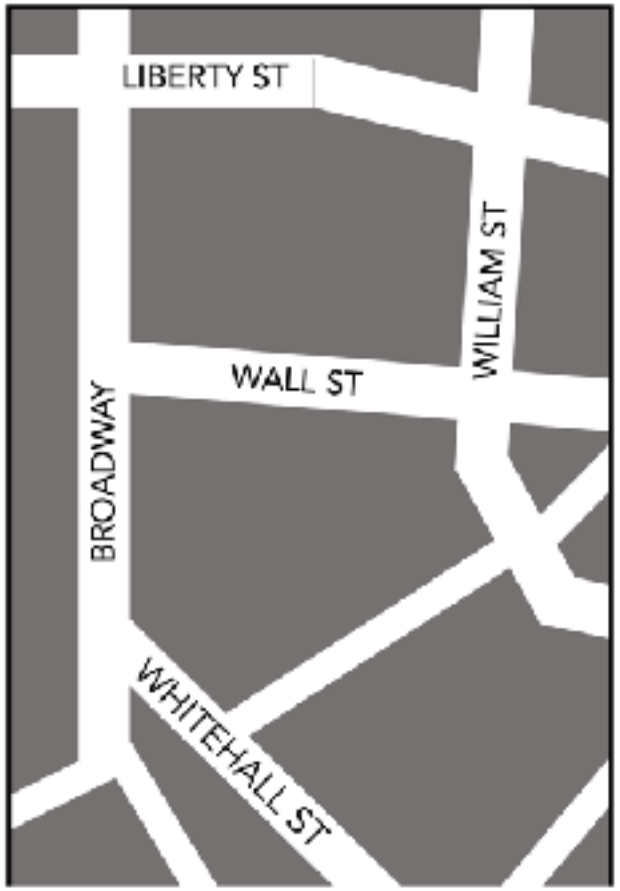
Wang et. al. PLDI 2017

Bongard problem 47



Bongard, M. (1970). *Pattern recognition*.

An Example



GreenSignal

Broadway

Liberty St

William St

Whitehall St

HasTraffic

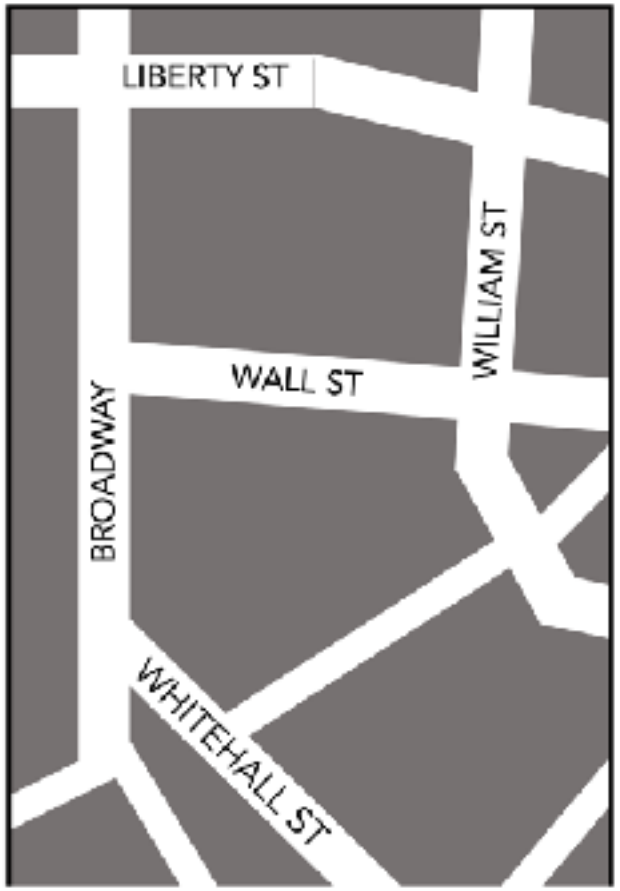
Broadway

Wall St

William St

Whitehall St

An Example



GreenSignal

Broadway
Liberty St
William St
Whitehall St

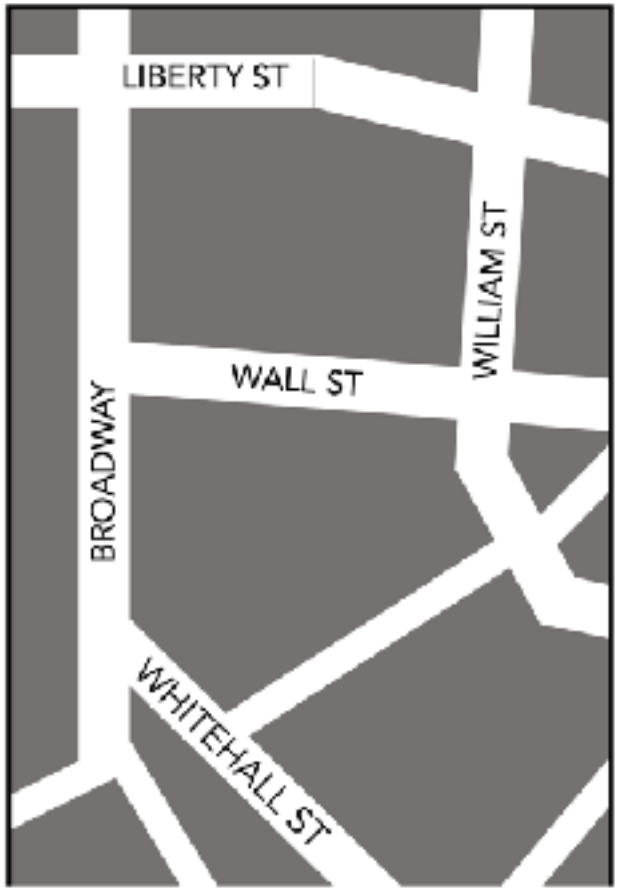
HasTraffic

Broadway
Wall St
William St
Whitehall St

Crashes

Broadway
Whitehall St

An Example

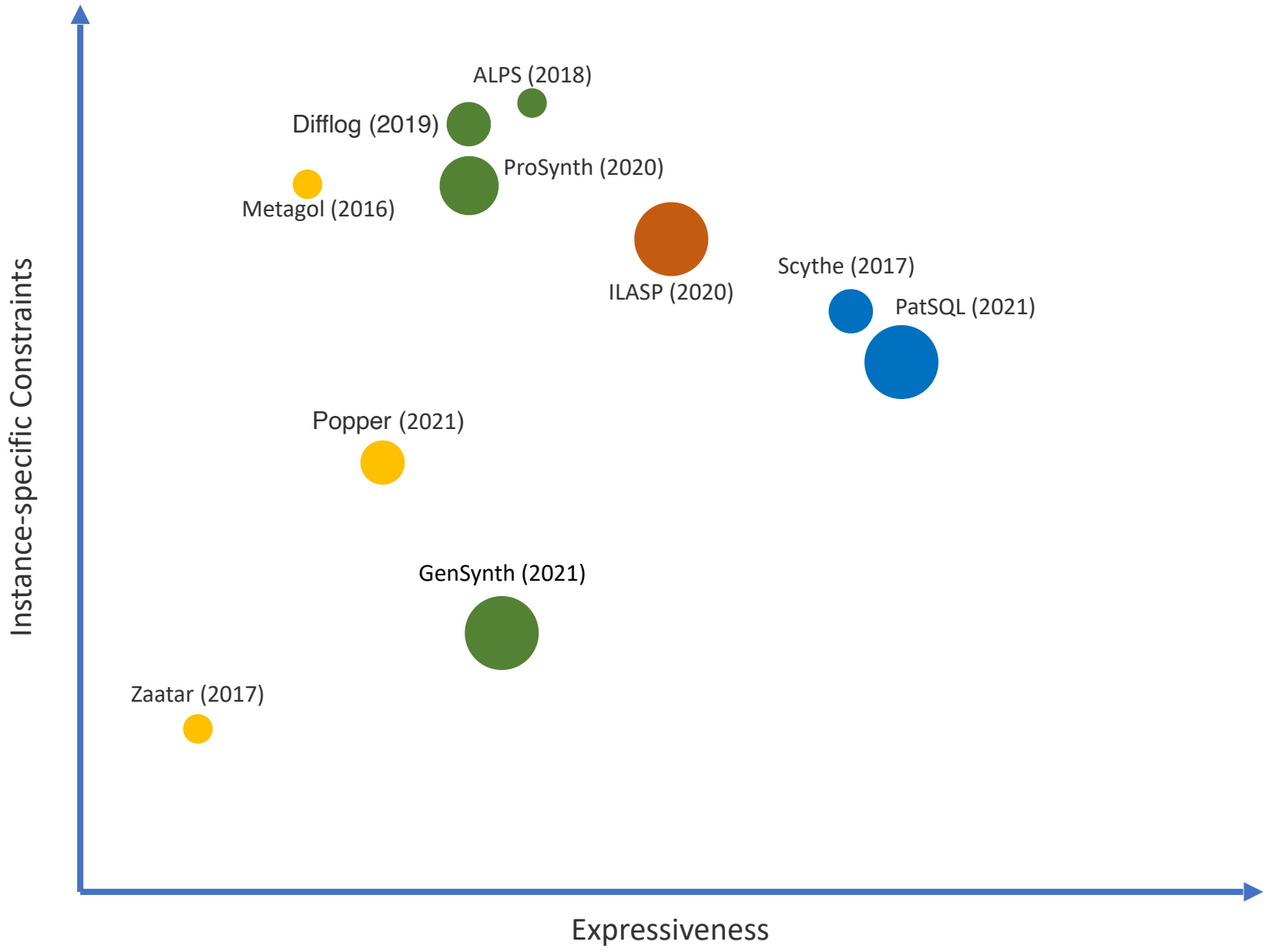


GreenSignal
Broadway
Liberty St
William St
Whitehall St

HasTraffic
Broadway
Wall St
William St
Whitehall St

Crashes
Broadway
Whitehall St

Crashes(x) :– HasTraffic(x), isGreen(x),
Intersects(x, y),
HasTraffic(y), isGreen(y).

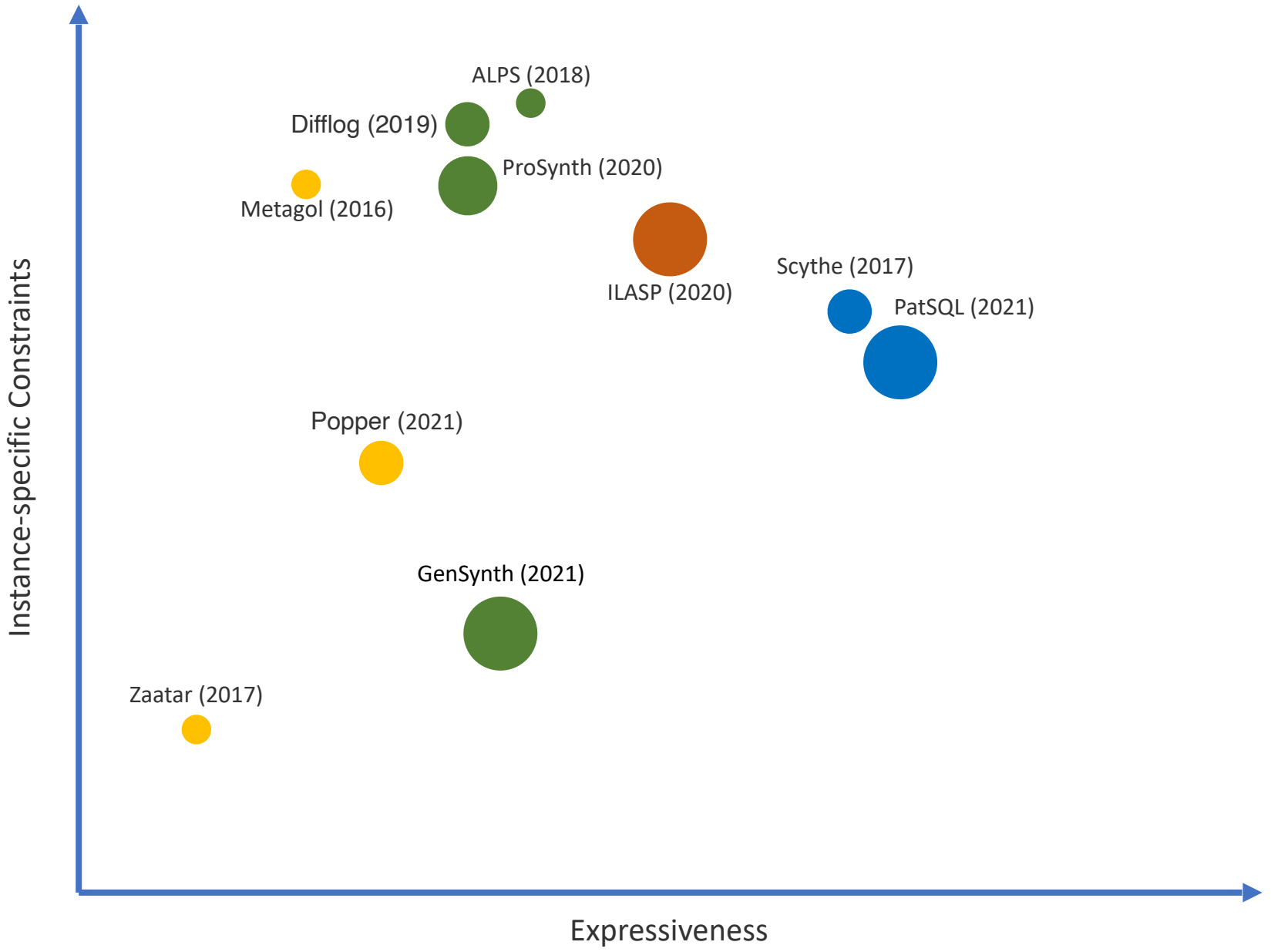

ALPS (2018)
Difflog (2019)
ProSynth (2020)
Metagol (2016)
ILASP (2020)
Scythe (2017)
PatSQL (2021)
Popper (2021)
GenSynth (2021)
Zaatar (2017)
Instance-specific Constraints
Expressiveness

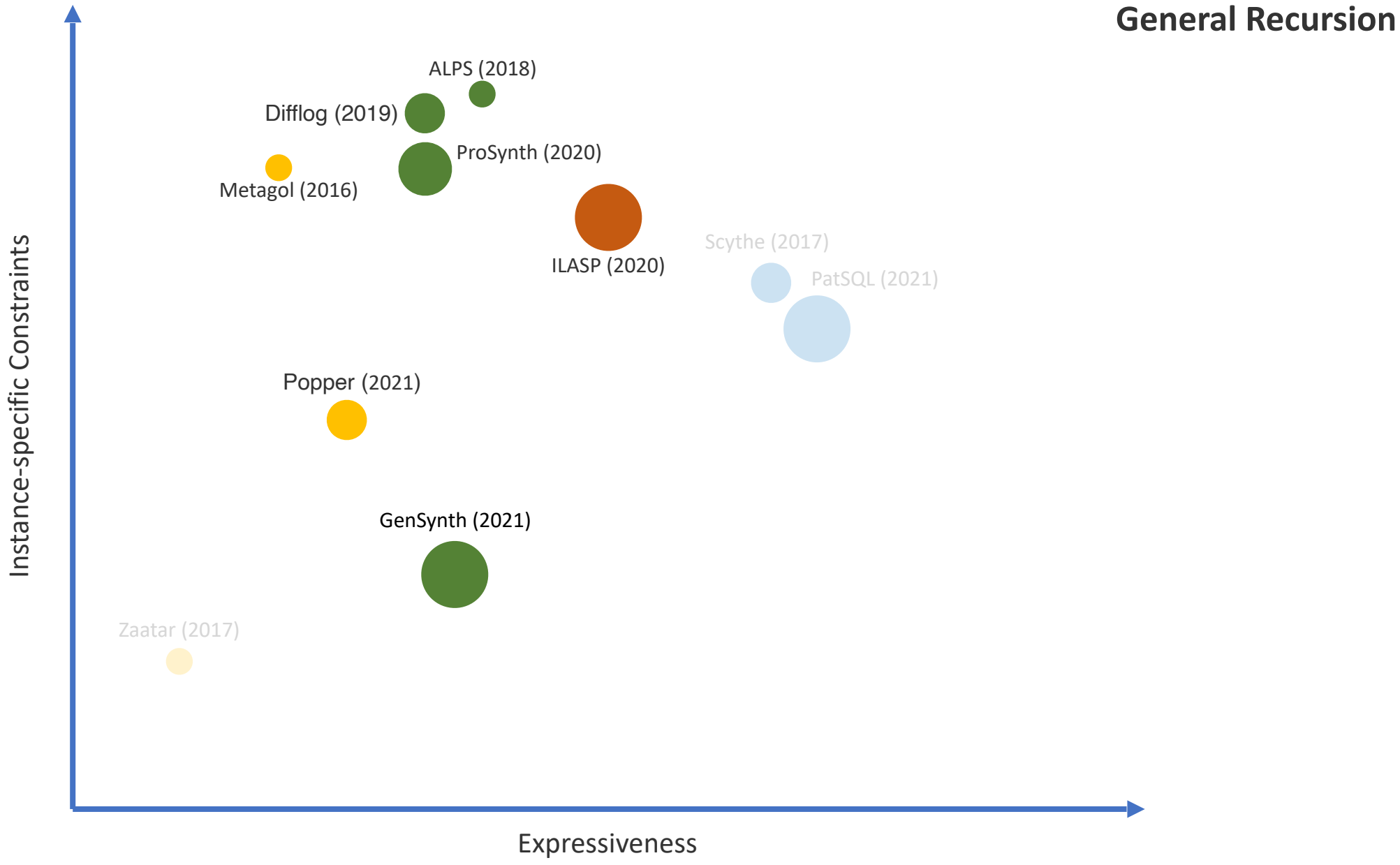
Templates

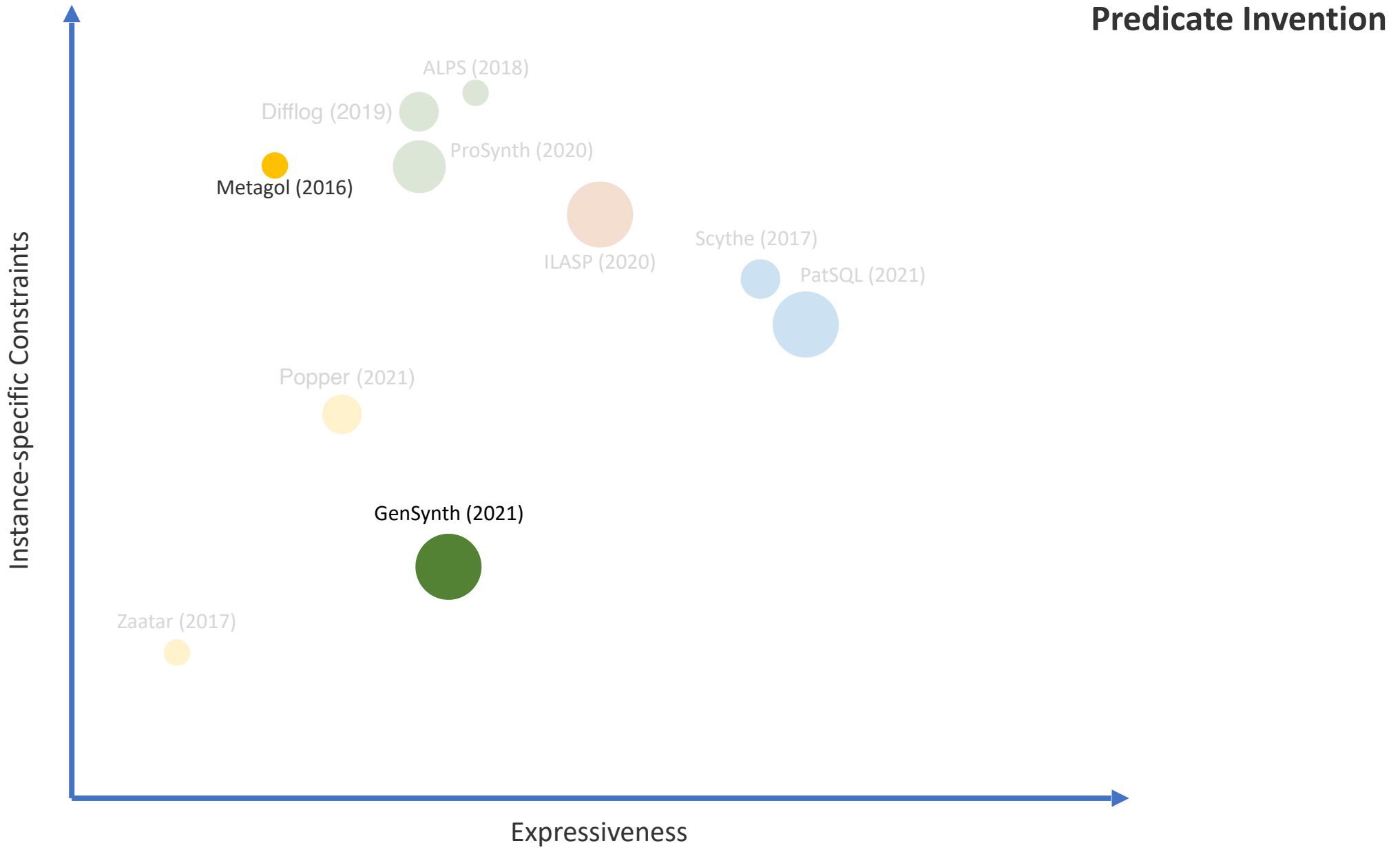
Meta Rules

Mode Declarations

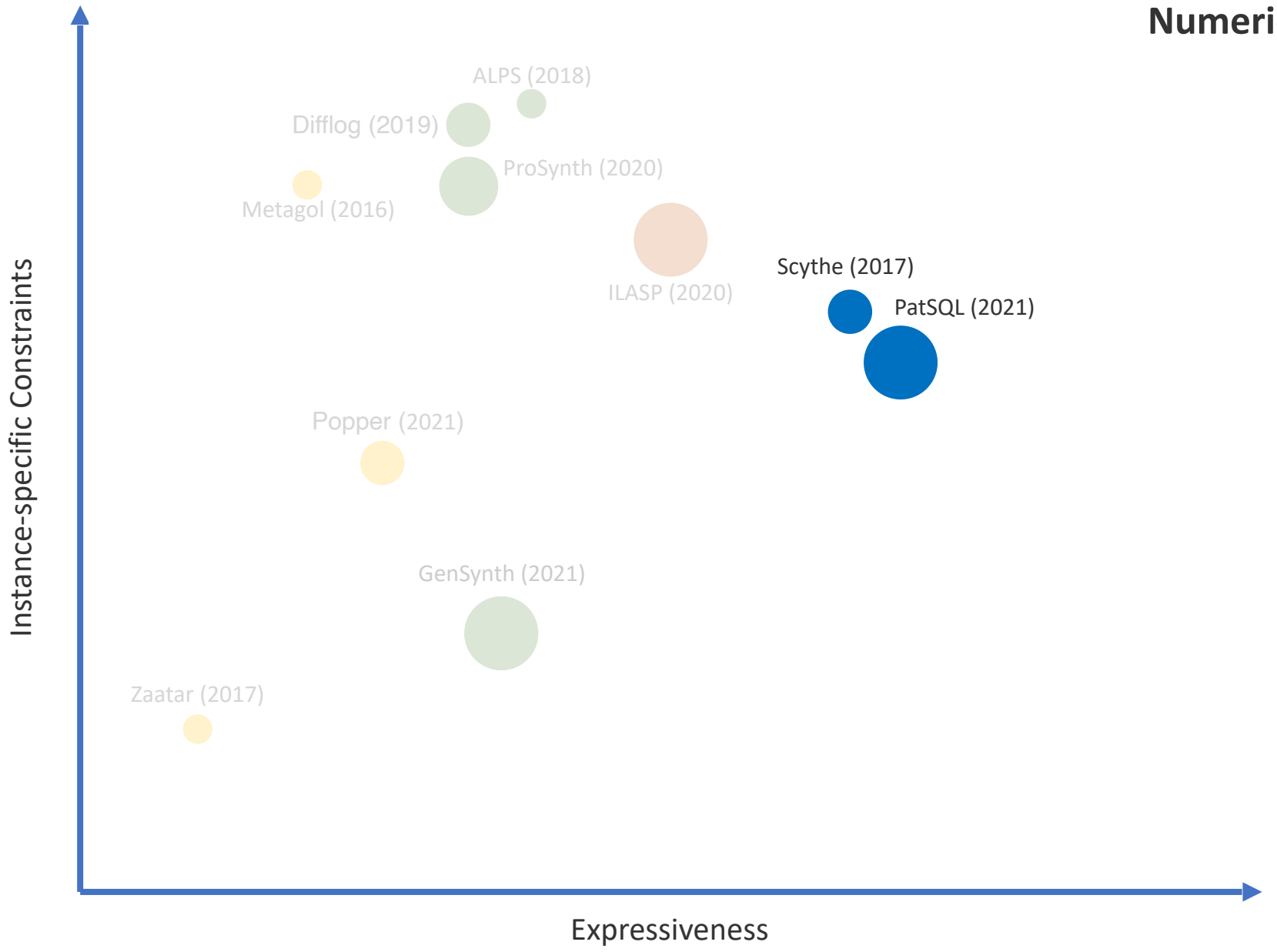
Comparison Constants

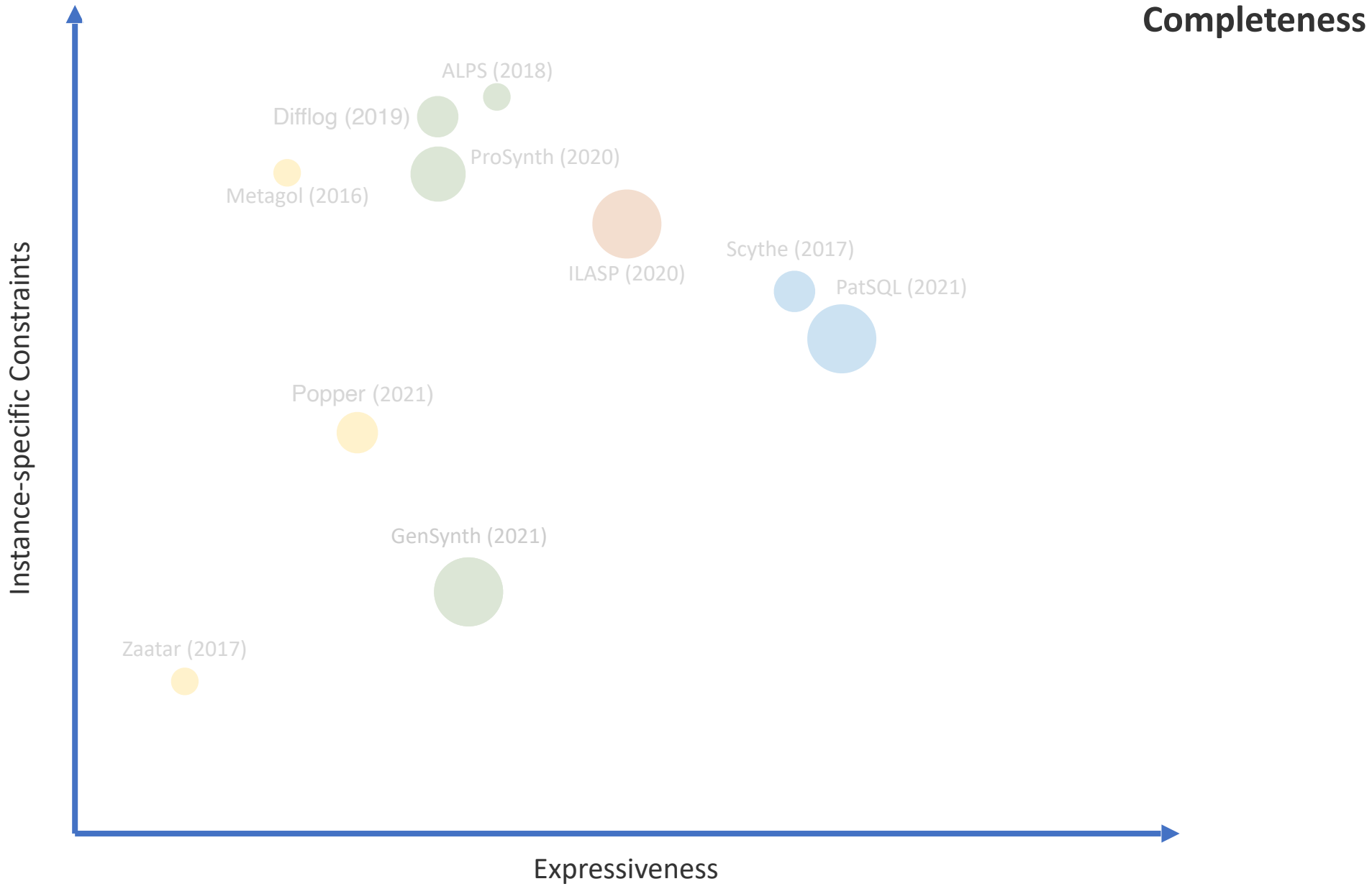


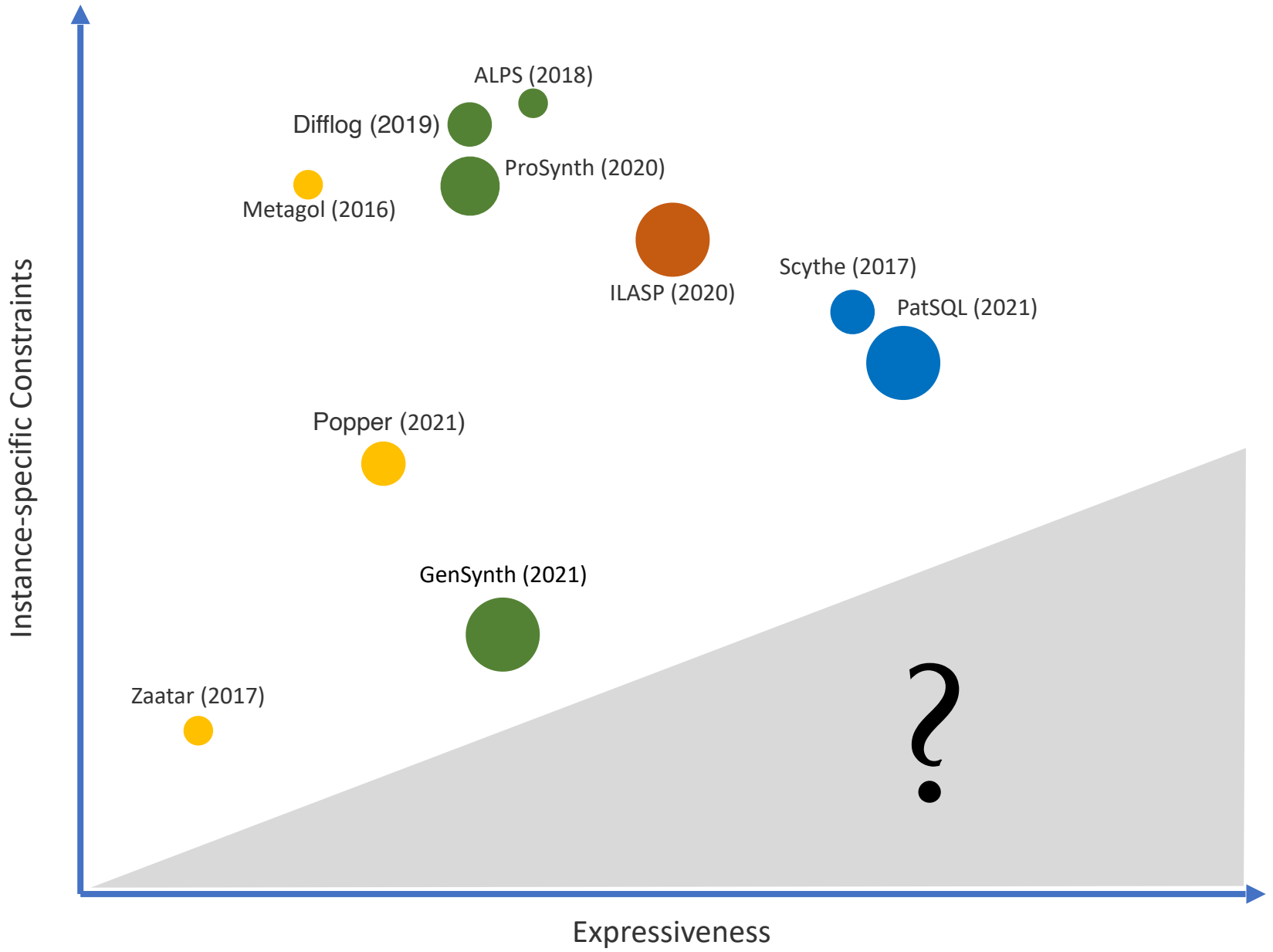

General Recursion
ALPS (2018)
Difflog (2019)
ProSynth (2020)
Metagol (2016)
ILASP (2020)
Scythe (2017)
PatSQL (2021)
Popper (2021)
GenSynth (2021)
Zaatar (2017)
Instance-specific Constraints
Expressiveness


Predicate Invention
ALPS (2018)
Difflog (2019)
ProSynth (2020)
Metagol (2016)
ILASP (2020)
Scythe (2017)
PatSQL (2021)
Popper (2021)
GenSynth (2021)
Zaatar (2017)
Instance-specific Constraints
Expressiveness

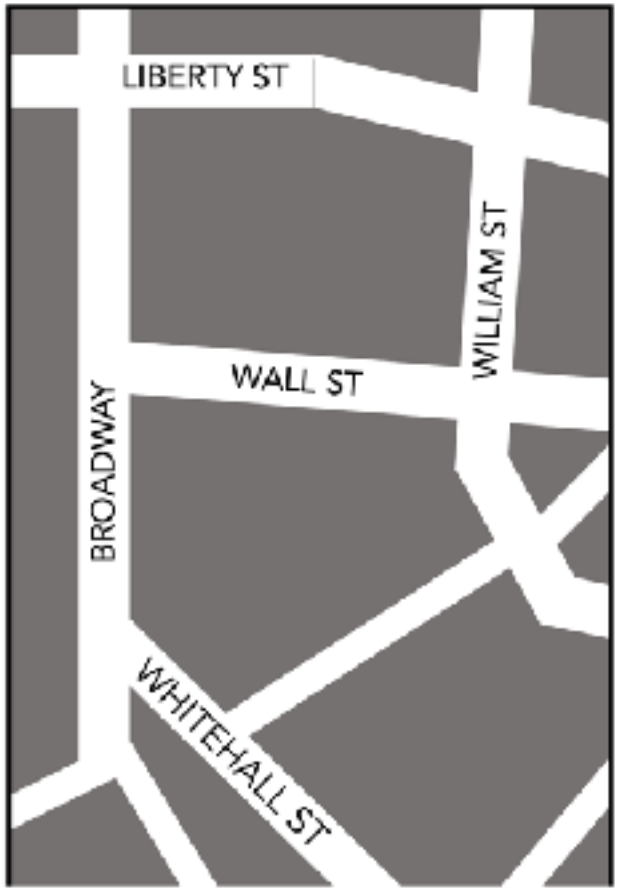
Numerical Comparison




Completeness
ALPS (2018)
Difflog (2019)
ProSynth (2020)
Metagol (2016)
ILASP (2020)
Scythe (2017)
PatSQL (2021)
Popper (2021)
GenSynth (2021)
Zaatar (2017)
Instance-specific Constraints
Expressiveness


ALPS (2018)
Difflog (2019)
ProSynth (2020)
Metagol (2016)
ILASP (2020)
Scythe (2017)
PatSQL (2021)
Popper (2021)
GenSynth (2021)
Zaatar (2017)
?
Instance-specific Constraints
Expressiveness

Decidability and Complexity

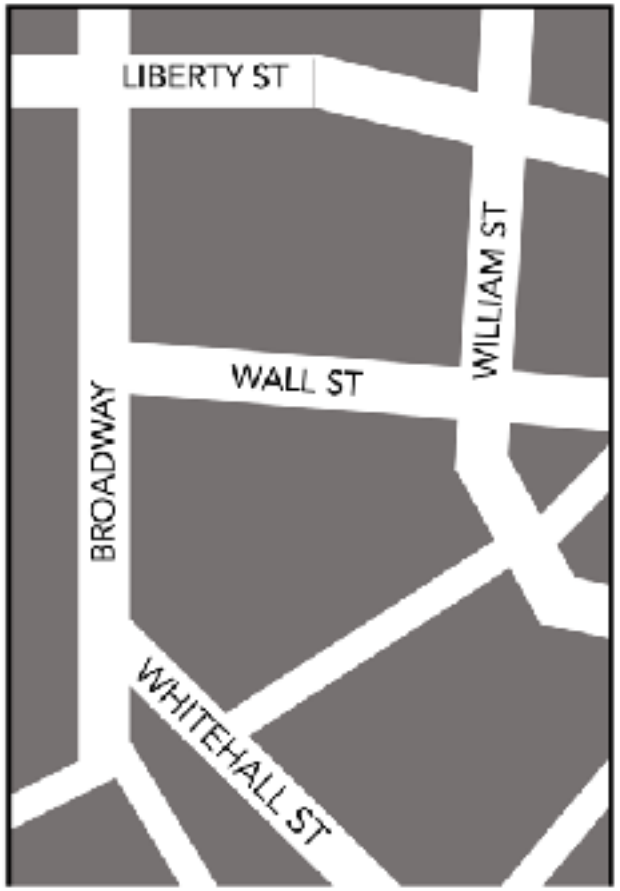


GreenSignal
Broadway
Liberty St
William St
Whitehall St

HasTraffic
Broadway
Wall St
William St
Whitehall St

Crashes
Broadway
Whitehall St

Decidability and Complexity



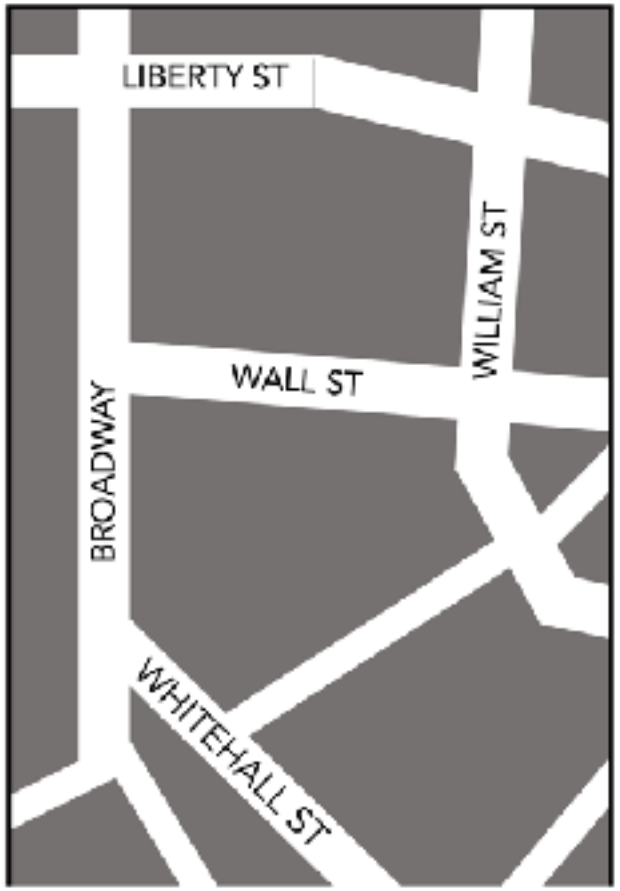
GreenSignal
Broadway
Liberty St
William St
Whitehall St

HasTraffic
Broadway
Wall St
William St
Whitehall St

Crashes
Broadway
Whitehall St

Constant-free Project-Join Queries

Decidability and Complexity



GreenSignal
Broadway
Liberty St
William St
Whitehall St

HasTraffic
Broadway
Wall St
William St
Whitehall St

Crashes
Broadway
Whitehall St

Conjunction of all the facts

Conclusion

Decidability and Complexity

Intersects(Liberty, Broadway),
Intersects(Broadway, Liberty),
Intersects(Broadway, Wall),
Intersects(Wall, Broadway),
Intersects(Liberty, William),
Intersects(William, Liberty),
Intersects(William, Wall),
Intersects(Wall, William),
Intersects(Whitehall, Broadway),
Intersects(Broadway, Whitehall)

GreenSignal(Broadway),
GreenSignal(Liberty),
GreenSignal(William),
GreenSignal(Whitehall)

HasTraffic(Broadway),
HasTraffic(Wall),
HasTraffic(William),
HasTraffic(Whitehall)

Crashes
Broadway
Whitehall St

Conjunction of all the facts

Conclusion

Decidability and Complexity

Intersects(Liberty, Broadway),
Intersects(Broadway, Liberty),
Intersects(Broadway, Wall),
Intersects(Wall, Broadway),
Intersects(Liberty, William),
Intersects(William, Liberty),
Intersects(William, Wall),
Intersects(Wall, William),
Intersects(Whitehall, Broadway),
Intersects(Broadway, Whitehall)

GreenSignal(Broadway),
GreenSignal(Liberty),
GreenSignal(William),
GreenSignal(Whitehall)

HasTraffic(Broadway),
HasTraffic(Wall),
HasTraffic(William),
HasTraffic(Whitehall)

Crashes
Broadway
Whitehall St

Abstraction of conjunction of all the facts

Conclusion

Decidability and Complexity

Crashes(x):

Intersects(x, y),
Intersects(y, x),
Intersects(y, z),
Intersects(z, y),
Intersects(x, w),
Intersects(w, x),
Intersects(w, z),
Intersects(z, w),
Intersects(u, y),
Intersects(y, u)

GreenSignal(y),
GreenSignal(x),
GreenSignal(w),
GreenSignal(u)

HasTraffic(y),
HasTraffic(z),
HasTraffic(w),
HasTraffic(u)

Crashes
Broadway
Whitehall St

Most specific constant-free project-join query

Decidability and Complexity

Crashes(x):

Intersects(w, x), GreenSignal(x),
Intersects(x, w), GreenSignal(w),
Intersects(x, u), GreenSignal(v),
Intersects(u, x), GreenSignal(y)
Intersects(w, v),
Intersects(v, w),
Intersects(v, u), HasTraffic(x),
Intersects(u, v), HasTraffic(u),
Intersects(y, x), HasTraffic(v),
Intersects(x, y) HasTraffic(y)

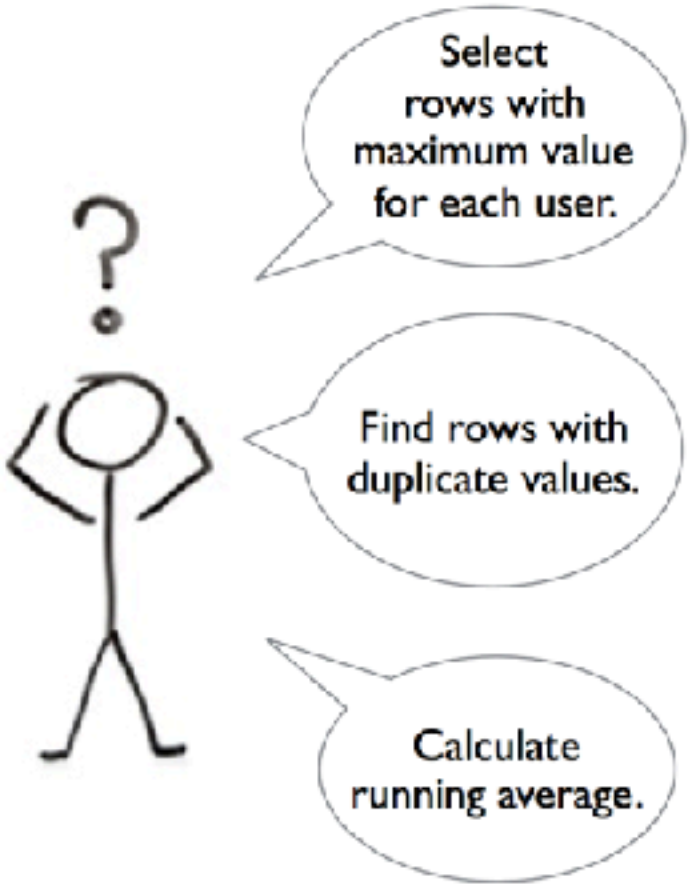
Crashes
Broadway
Whitehall St

Most specific constant-free project-join query

Decidability and Complexity

Most specific constant-free project-join query is consistent with the input-output examples if and only if the problem instance is realisable. Checking this is coNP complete.

End User



stackoverflow

SQL

```sql
Select x.id, x.customer, x.total
From PURCHASES x
Join (Select p.customer,
             Max(total)
      From PURCHASES p
      Group By p.customer) y
On y.customer = x.customer
   And y.max_total = x.total
```

```sql
Select *
From  Users a
Where Exists
   (Select *
    From Users b
    Where (a.name = b.name
           Or  a.email = b.email)
          And a.ID <> b.id)
```

```sql
Select a.ord, a.val, Avg(b.val)
From t As a Join t As b
Where b.ord <= a.ord
Group By a.ord,a.val
Order By a.ord
```

Wang et. al. PLDI 2017

Crashes(x):

Intersects(w, x),
Intersects(x, w),
Intersects(x, u),
Intersects(u, x),
Intersects(w, v),
Intersects(v, w),
Intersects(v, u),
Intersects(u, v),
Intersects(y, x),
Intersects(x, y)

GreenSignal(x),
GreenSignal(w),
GreenSignal(v),
GreenSignal(y)

HasTraffic(x),
HasTraffic(u),
HasTraffic(v),
HasTraffic(y)

Crashes(x):

Intersects(w, x),
Intersects(x, w),
Intersects(x, u),
Intersects(u, x),
Intersects(w, v),
Intersects(v, w),
Intersects(v, u),
Intersects(u, v),
Intersects(y, x),
Intersects(x, y)

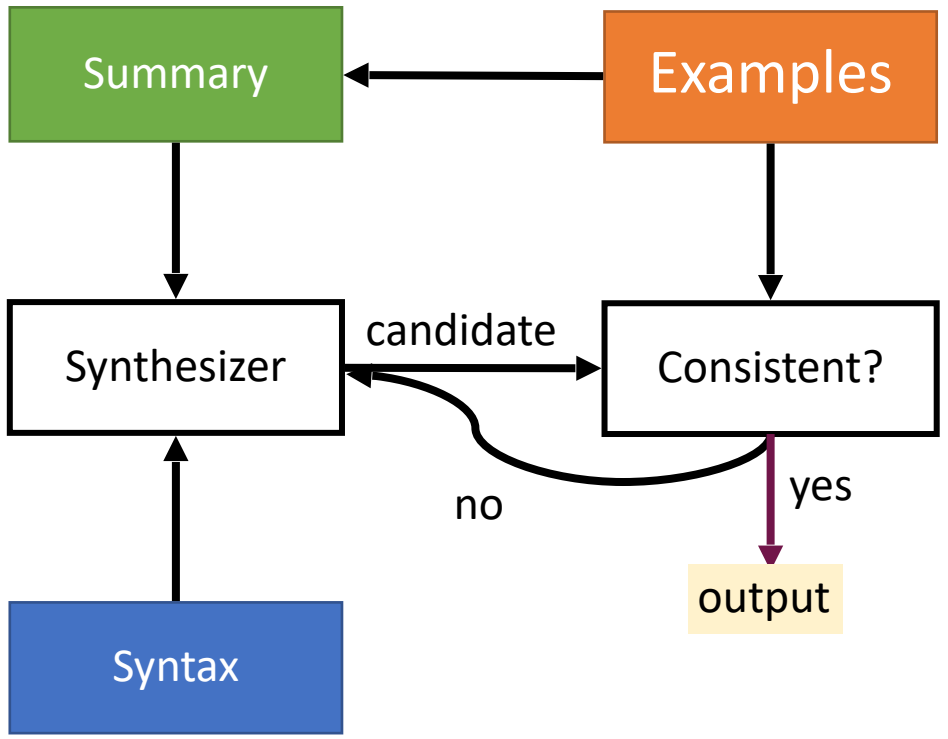
GreenSignal(x),
GreenSignal(w),
GreenSignal(v),
GreenSignal(y)

HasTraffic(x),
HasTraffic(u),
HasTraffic(v),
HasTraffic(y)

Example-Guided Synthesis of Relational Queries

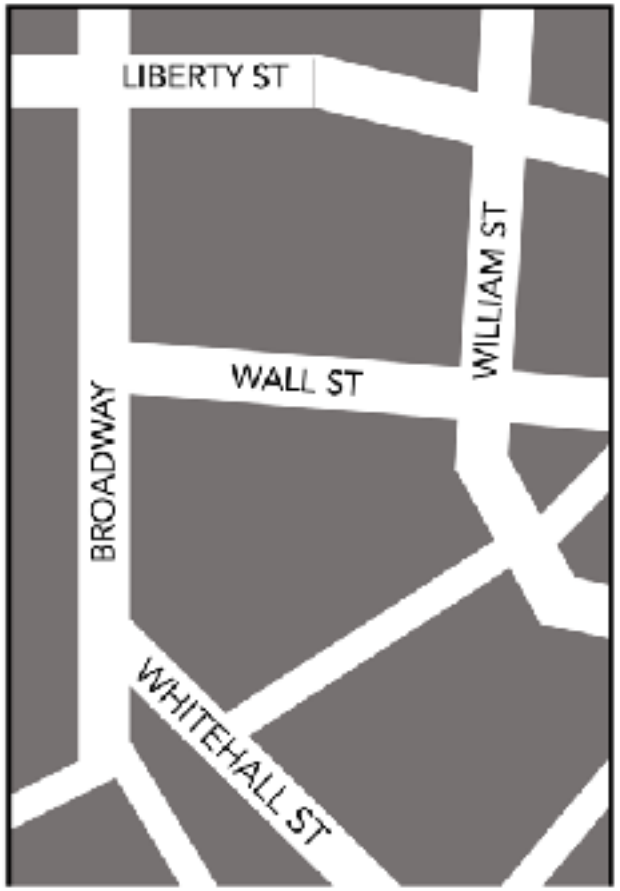
Example-guided Synthesis

1. Examples cannot be replaced by an evaluation oracle
2. Uses the latent *structure* of examples to generate the candidate programs
3. Outperforms syntax-guided techniques for relational queries



Example-Guided Synthesis of Relational Queries

An Example



GreenSignal

Broadway

Liberty St

William St

Whitehall St

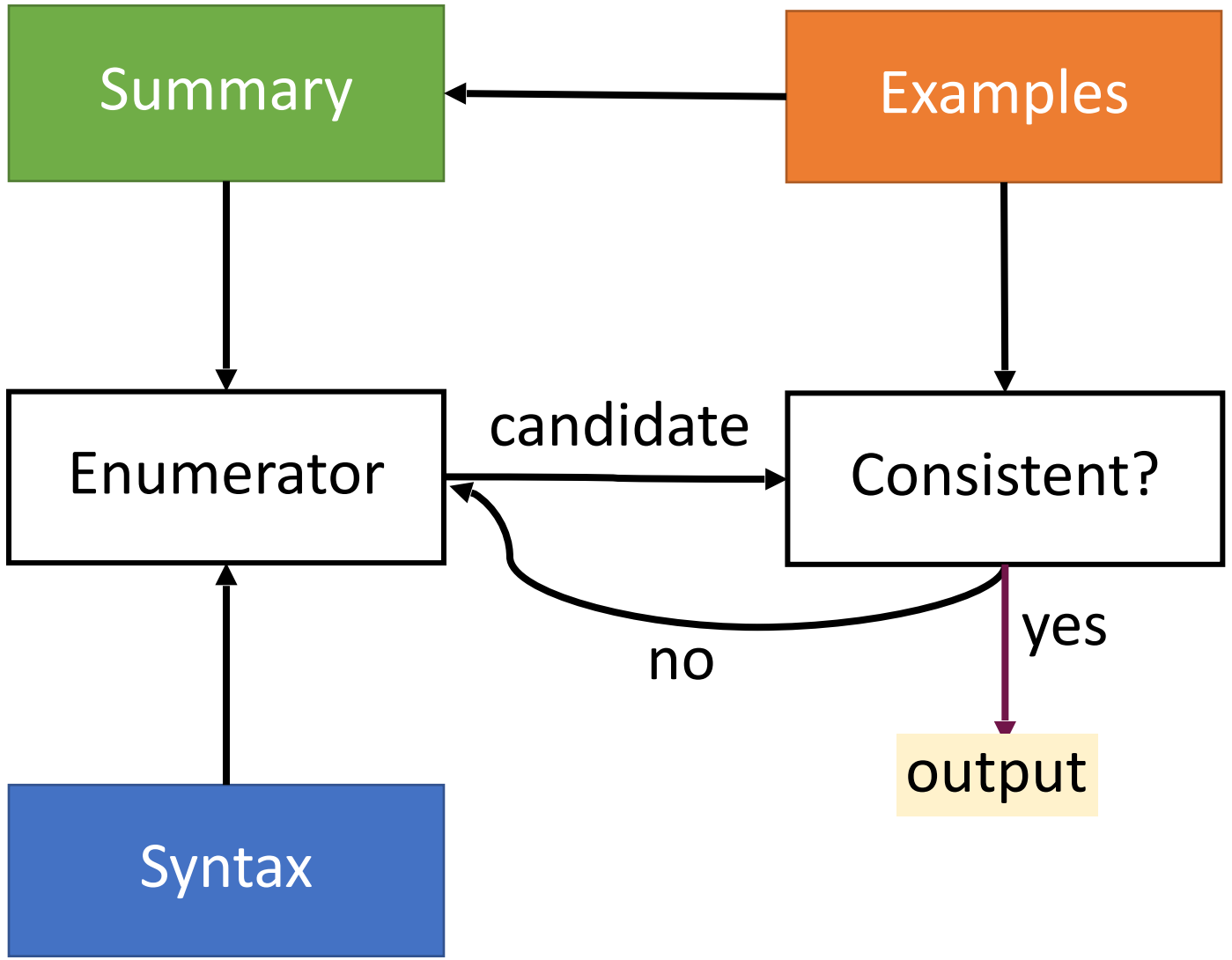
HasTraffic

Broadway

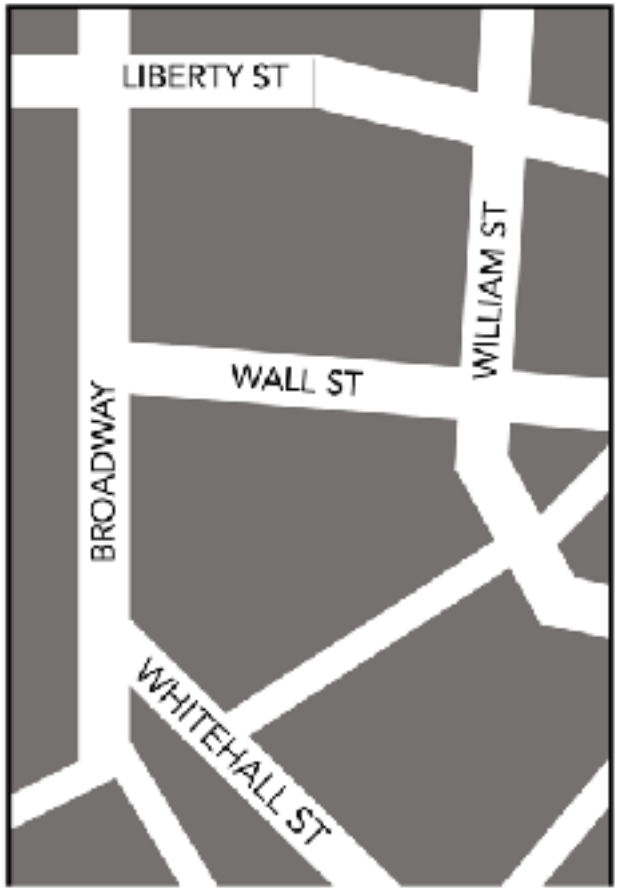
Wall St

William St

Whitehall St

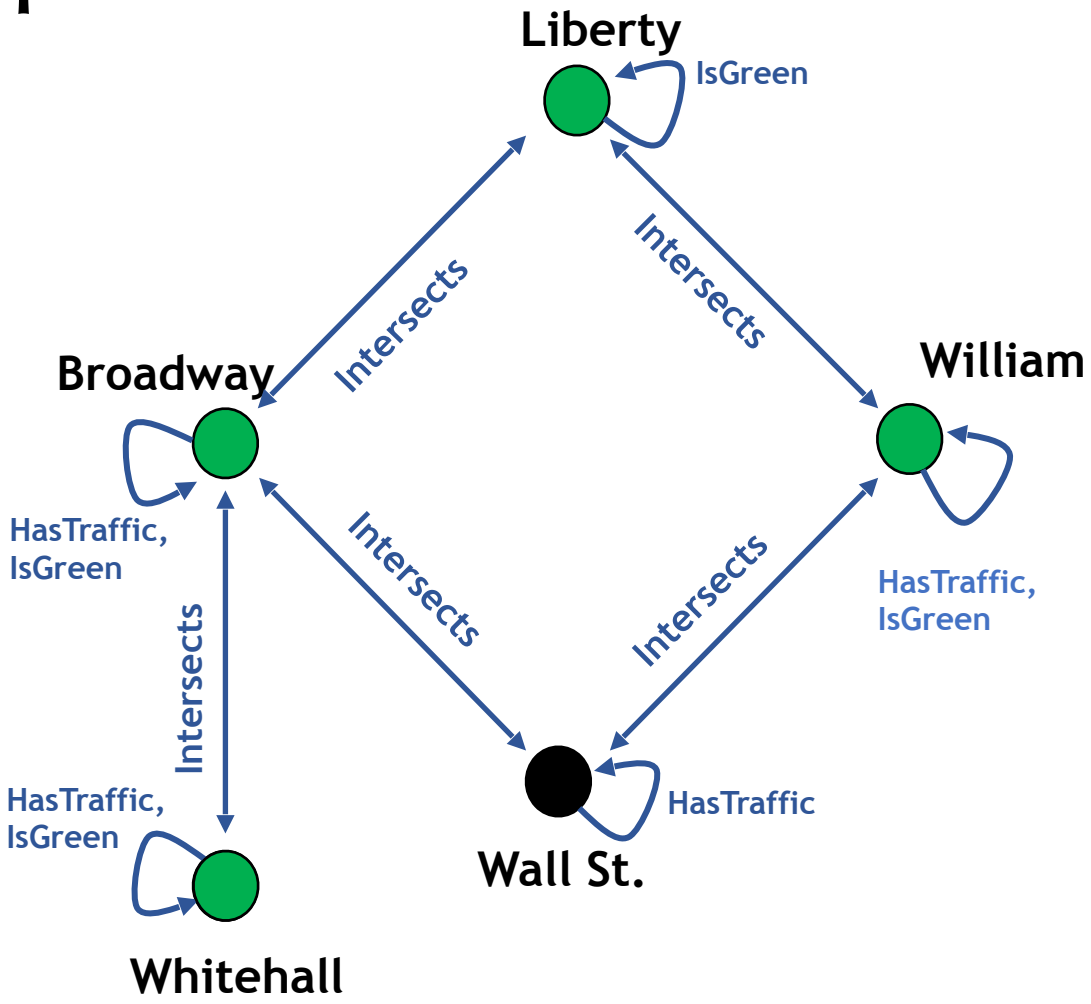

Summary
Examples
Enumerator
candidate
Consistent?
no
yes
output
Syntax

An Example



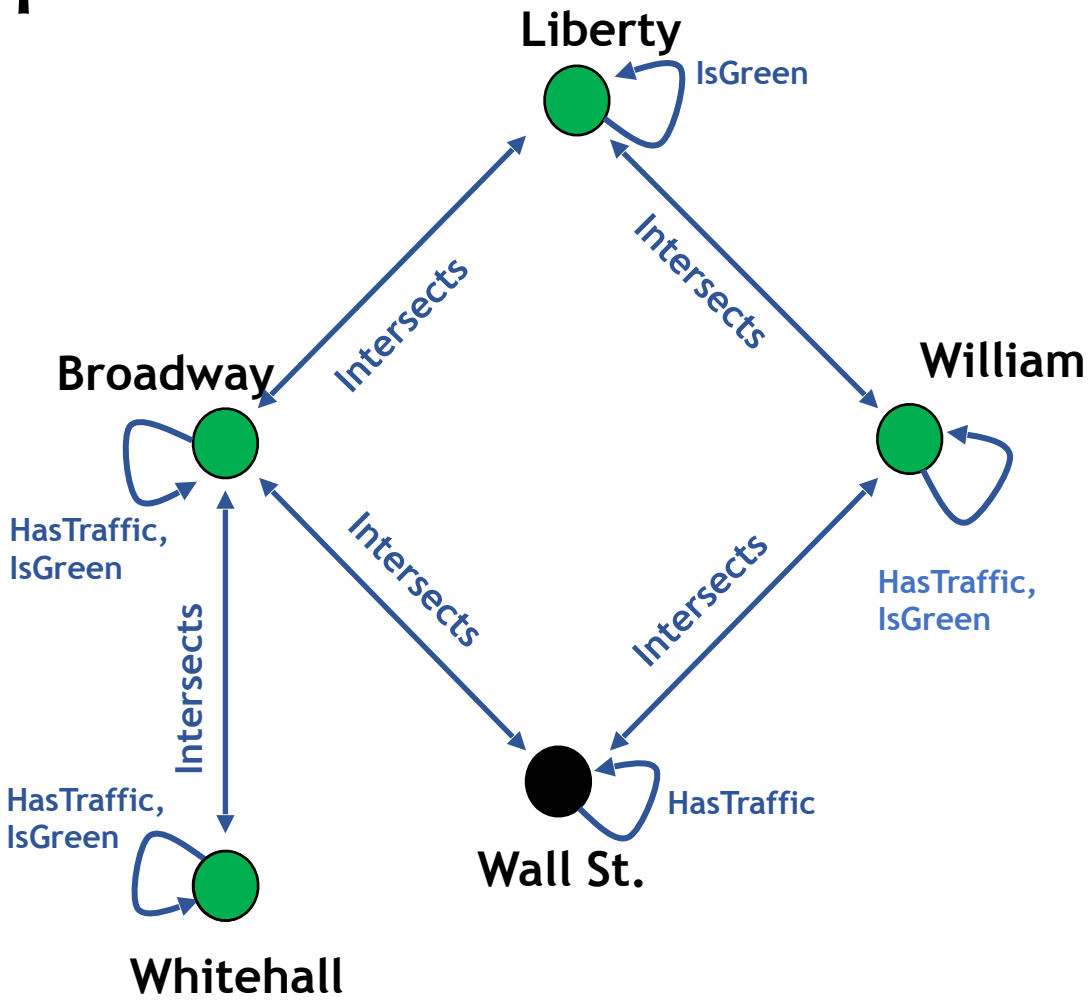
GreenSignal
Broadway
Liberty St
William St
Whitehall St

HasTraffic
Broadway
Wall St
William St
Whitehall St



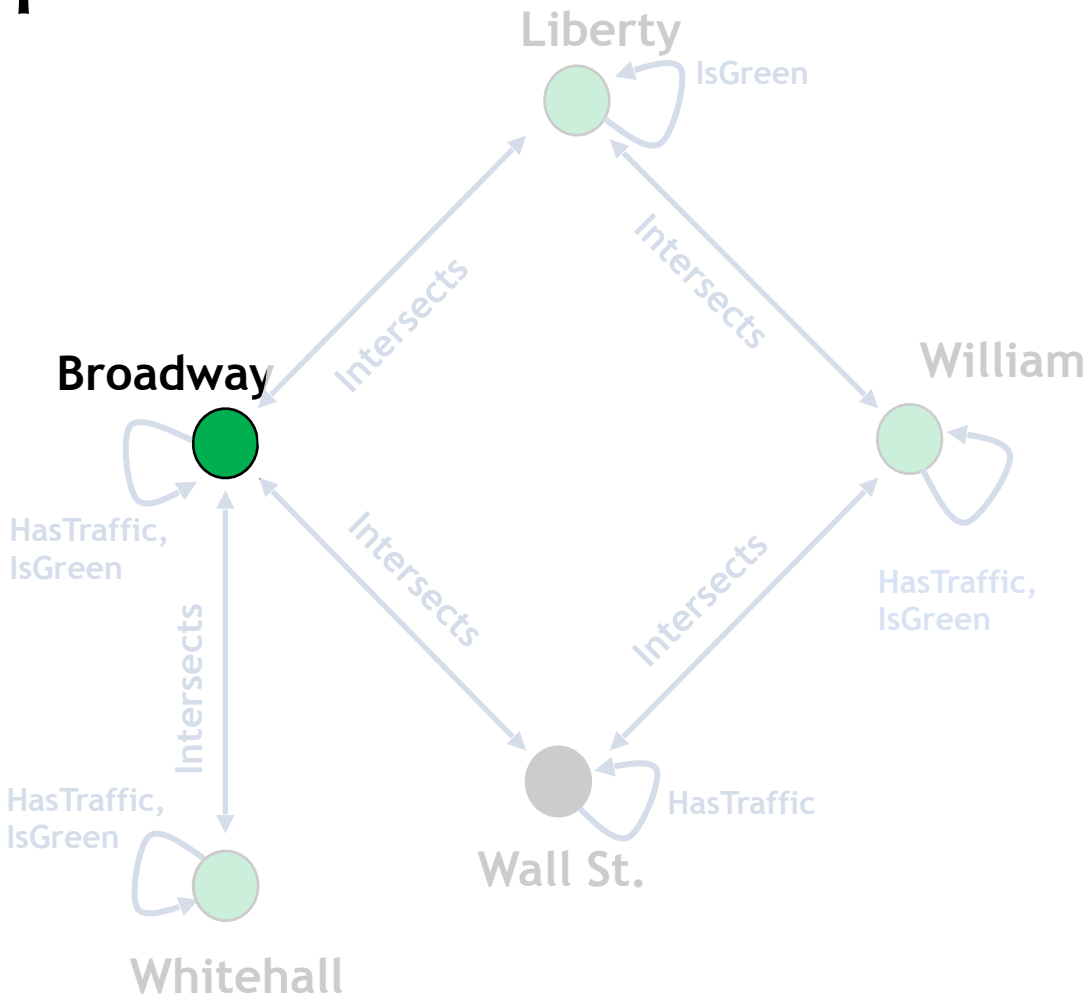
An Example

Crashes
Broadway
Whitehall St



An Example

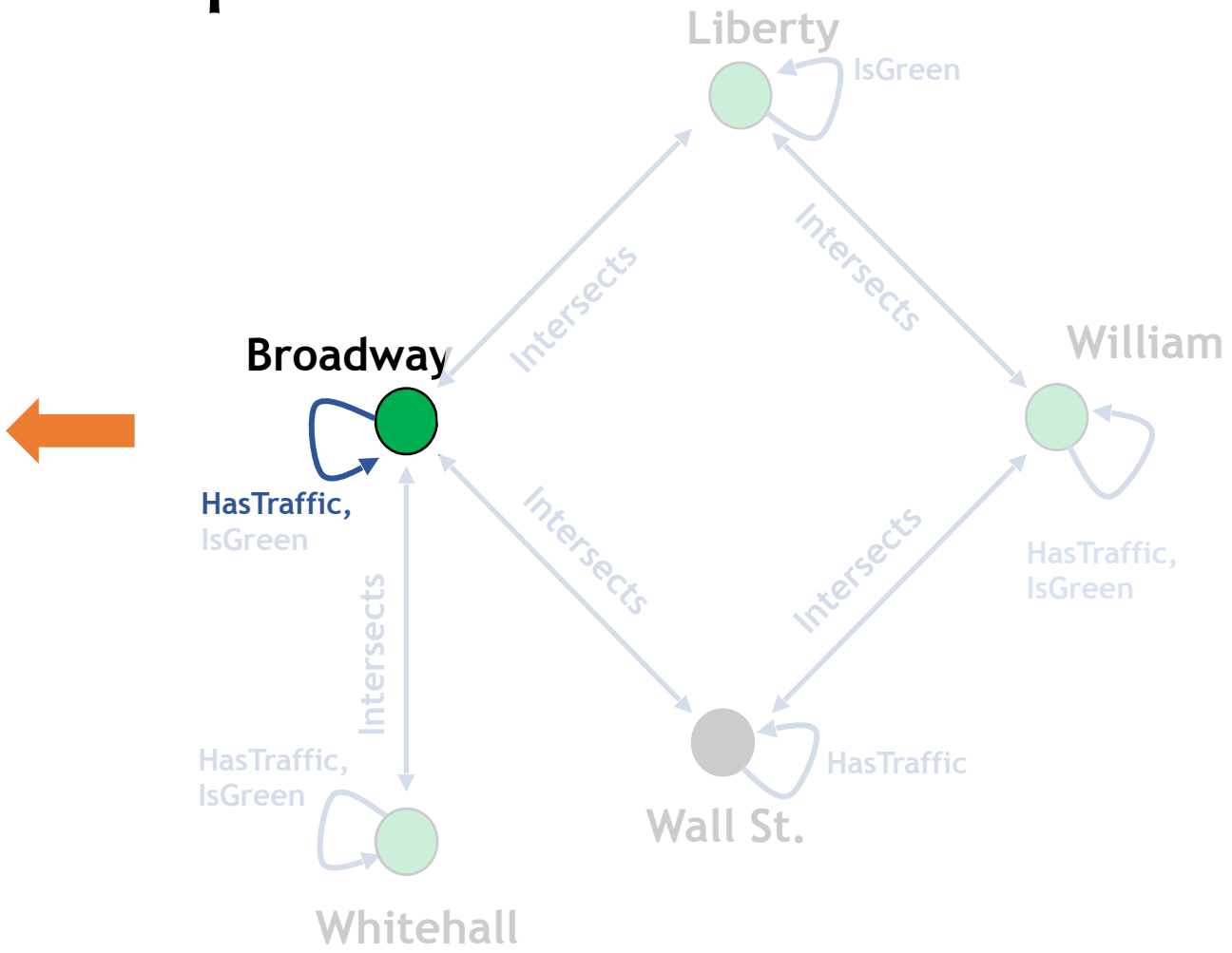
Crashes
Broadway
Whitehall St



An Example

Crashes
Broadway
Whitehall St

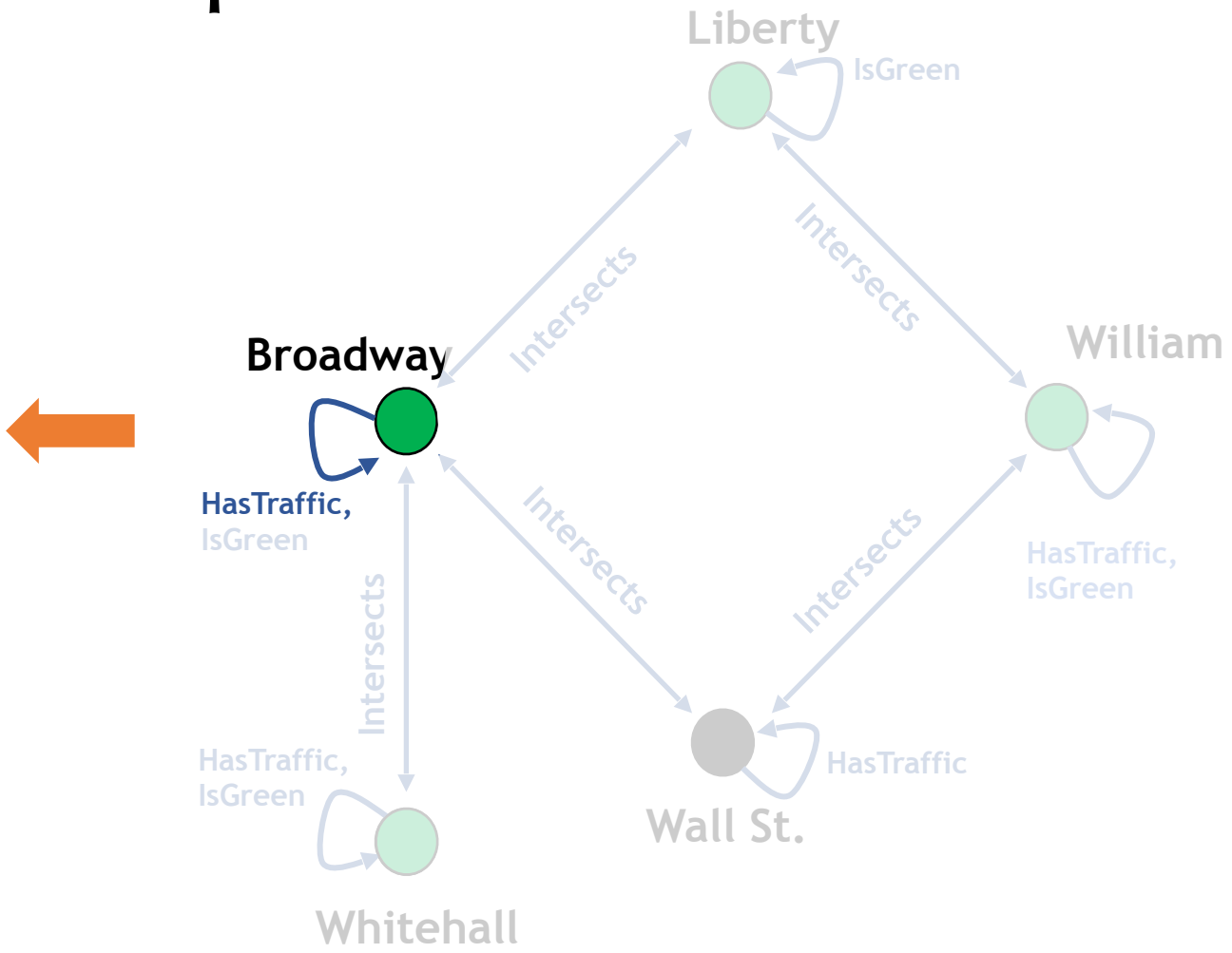
Crashes(Broadway) ← HasTraffic(Broadway).



An Example

Crashes
Broadway
Whitehall St

Crashes(x) :– HasTraffic(x).

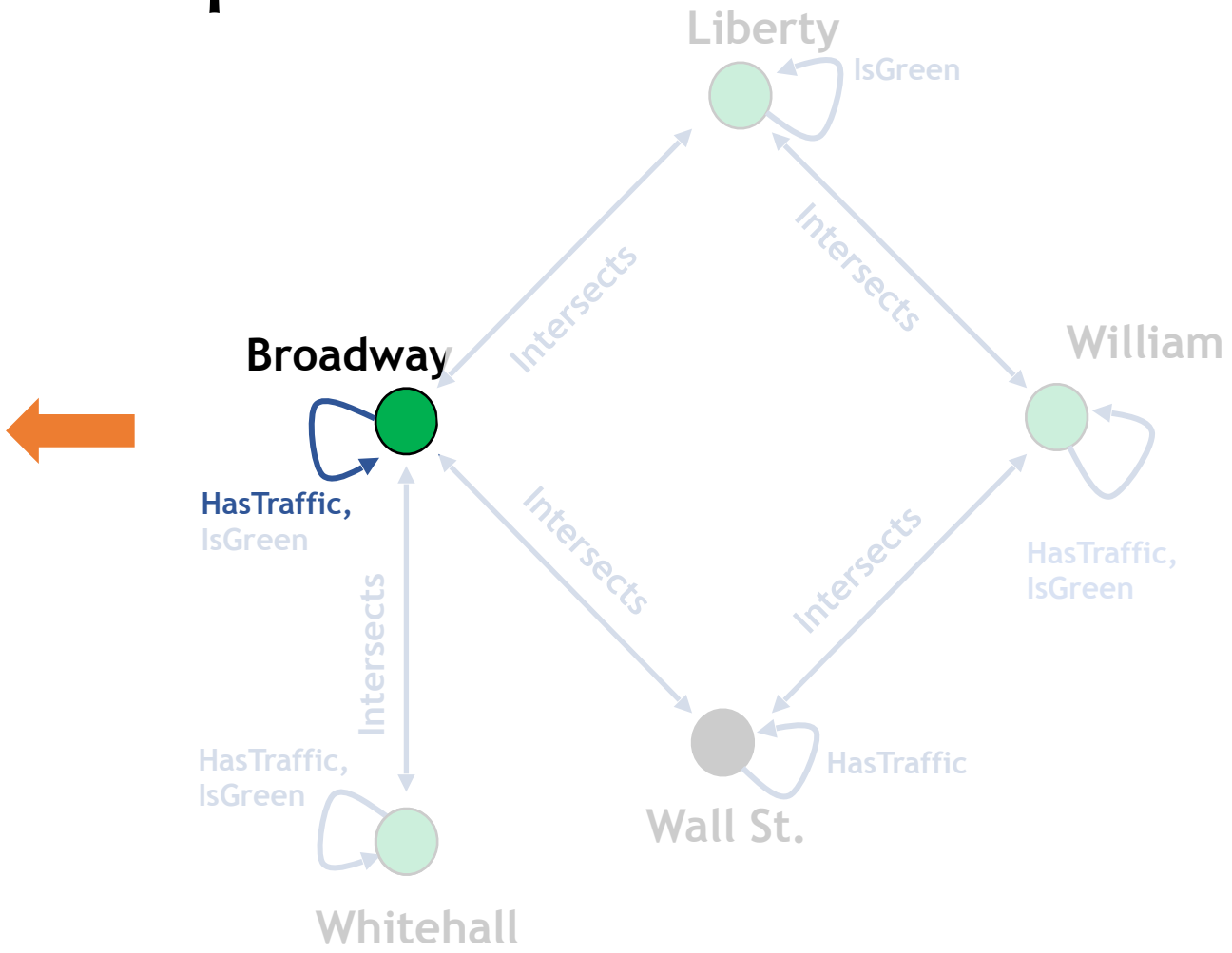


An Example

Crashes
Broadway
Whitehall St

Crashes(x) :– HasTraffic(x).

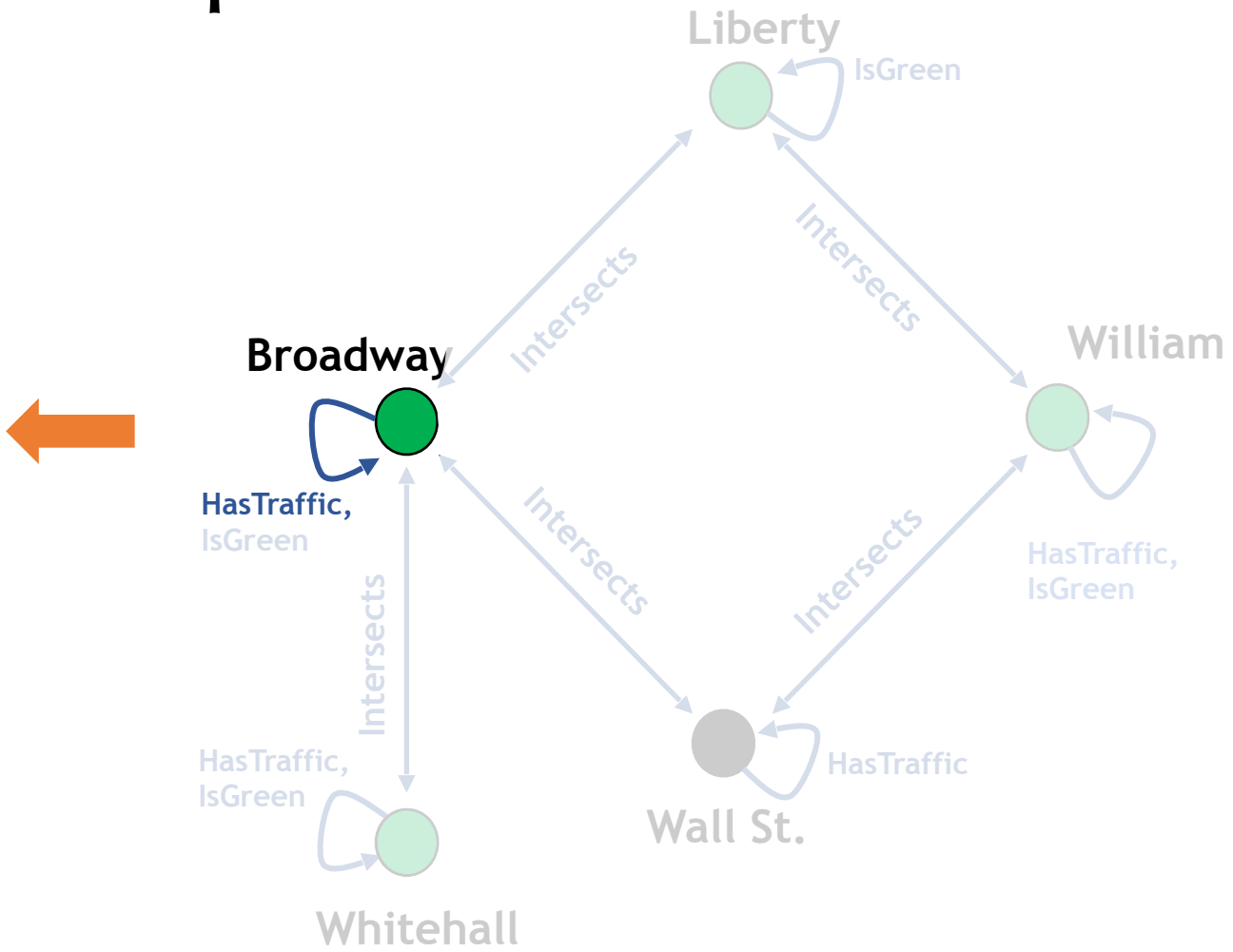
Crashes
Broadway
Wall St
William St
Whitehall St



An Example

Crashes
Broadway
Whitehall St

Crashes(x) :– HasTraffic(x).

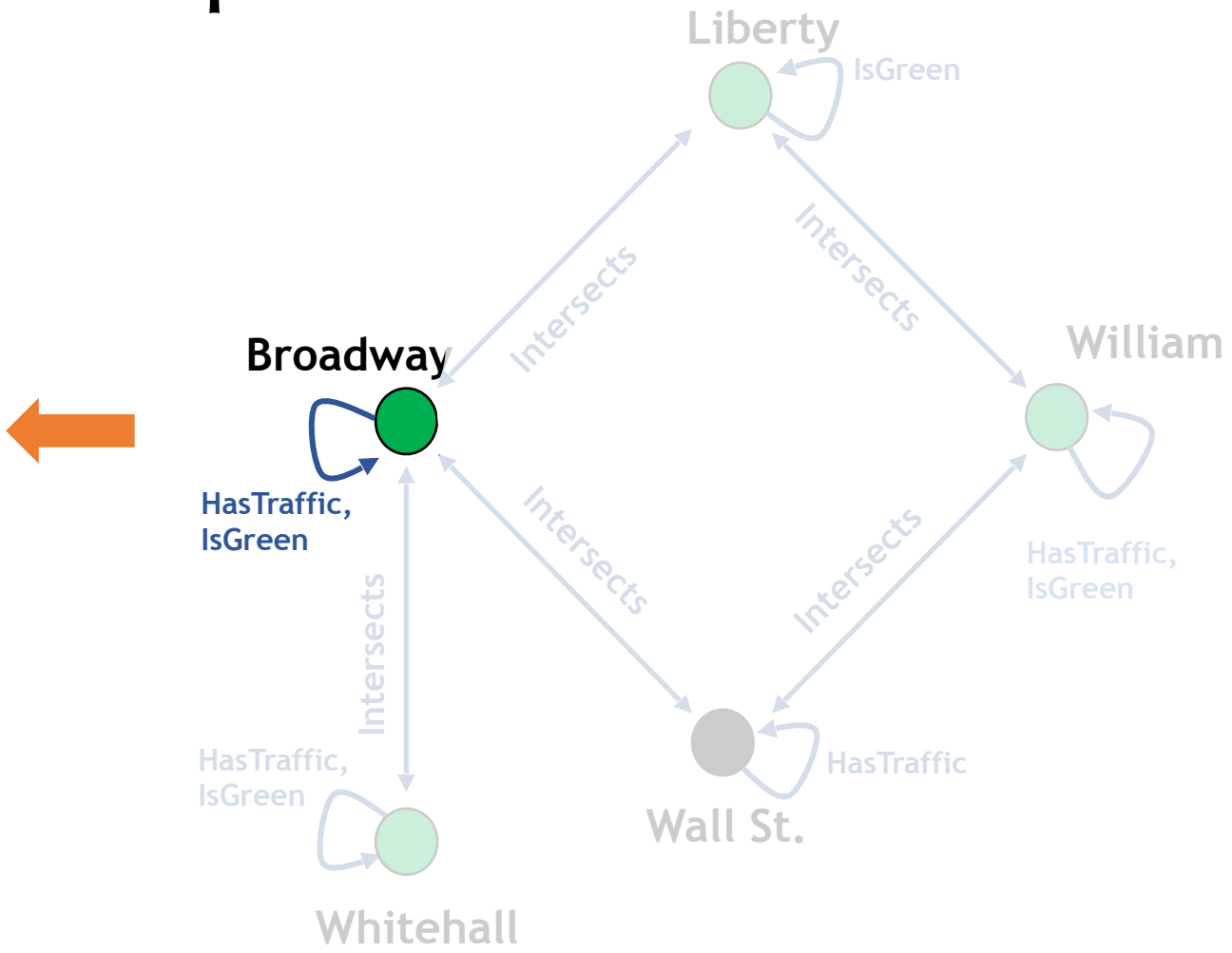


Crashes
Broadway
Wall St
William St
Whitehall St

An Example

Crashes
Broadway
Whitehall St

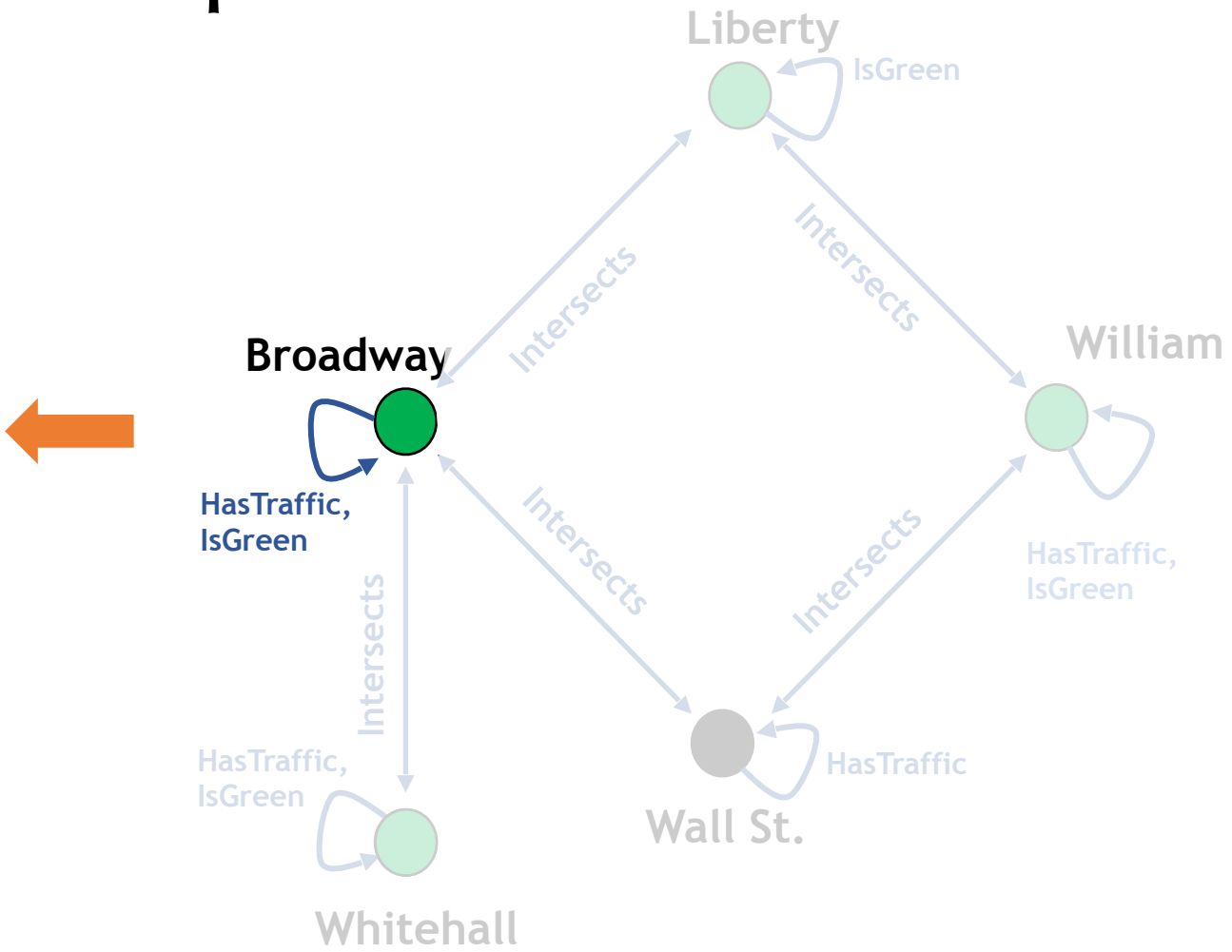
Crashes(x) :– HasTraffic(x), isGreen(x).



An Example

Crashes
Broadway
Whitehall St

Crashes(x) :– HasTraffic(x), isGreen(x).

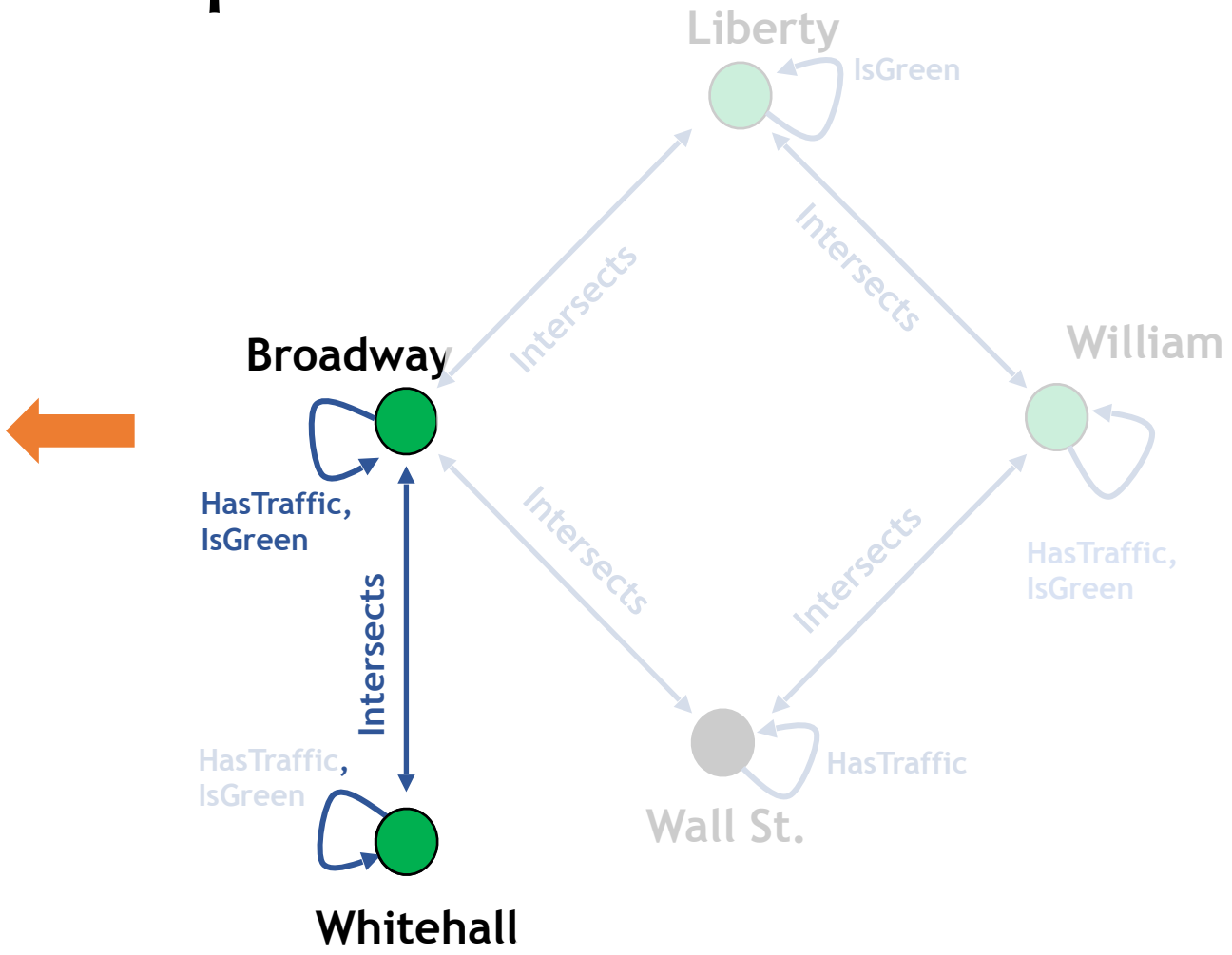


Crashes
Broadway
William St
Whitehall St

An Example

Crashes
Broadway
Whitehall St

Crashes(x) :– HasTraffic(x), isGreen(x),
Intersects(x, y).

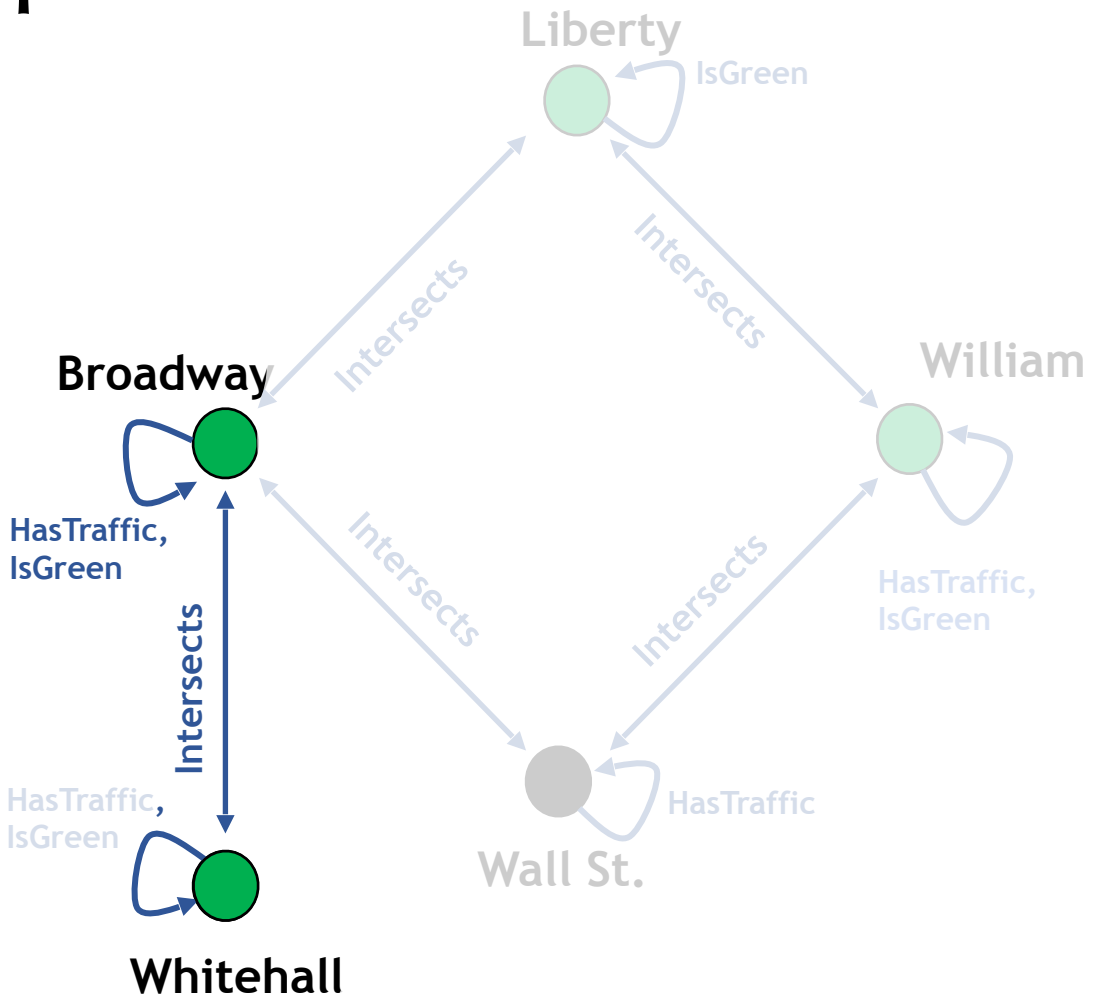


An Example

Crashes
Broadway
Whitehall St

Crashes(x) :– HasTraffic(x), isGreen(x),
Intersects(x, y).

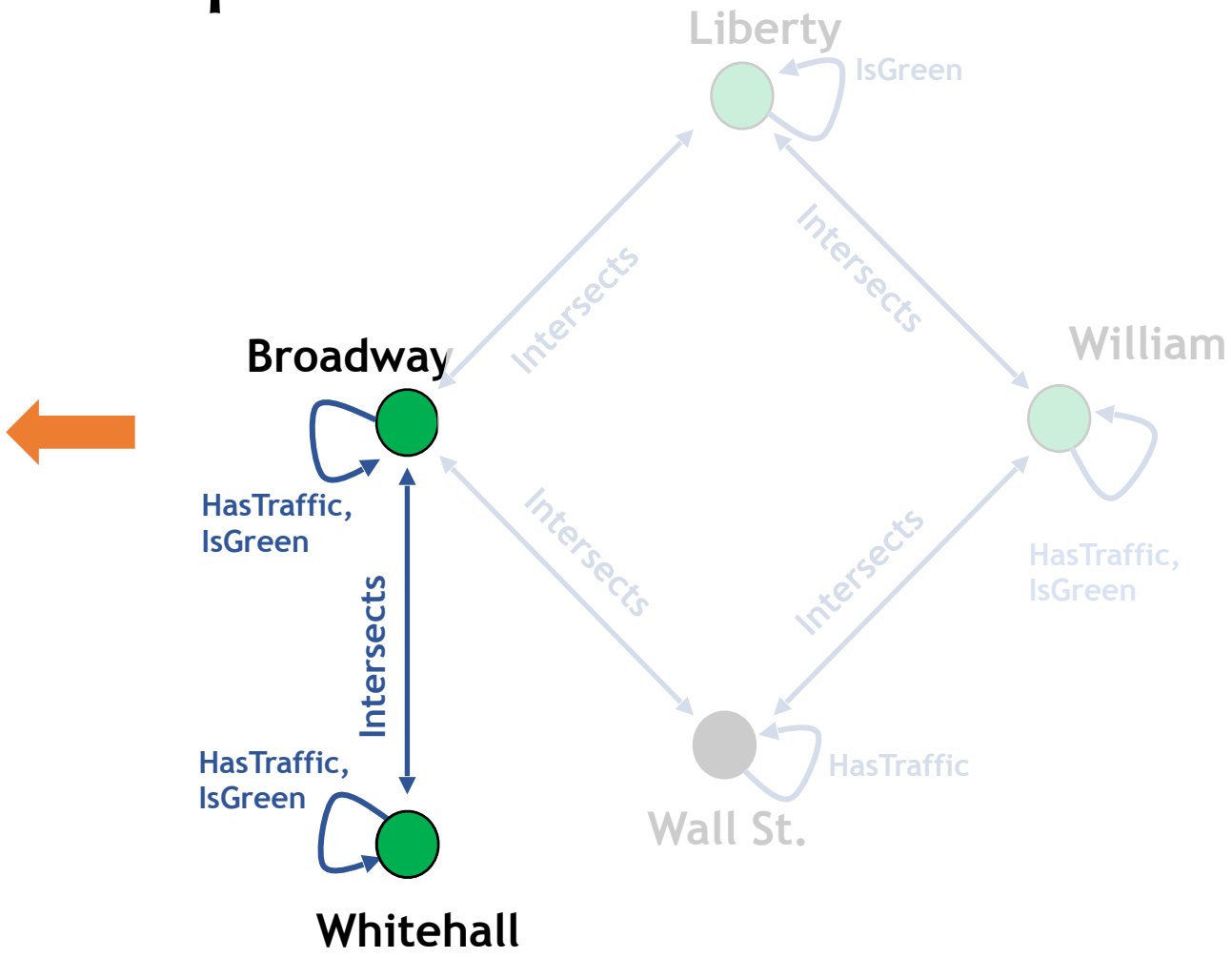
Crashes
Broadway
William St
Whitehall St



An Example

Crashes
Broadway
Whitehall St

Crashes(x) :– HasTraffic(x), isGreen(x),
Intersects(x, y),
HasTraffic(y), isGreen(y).

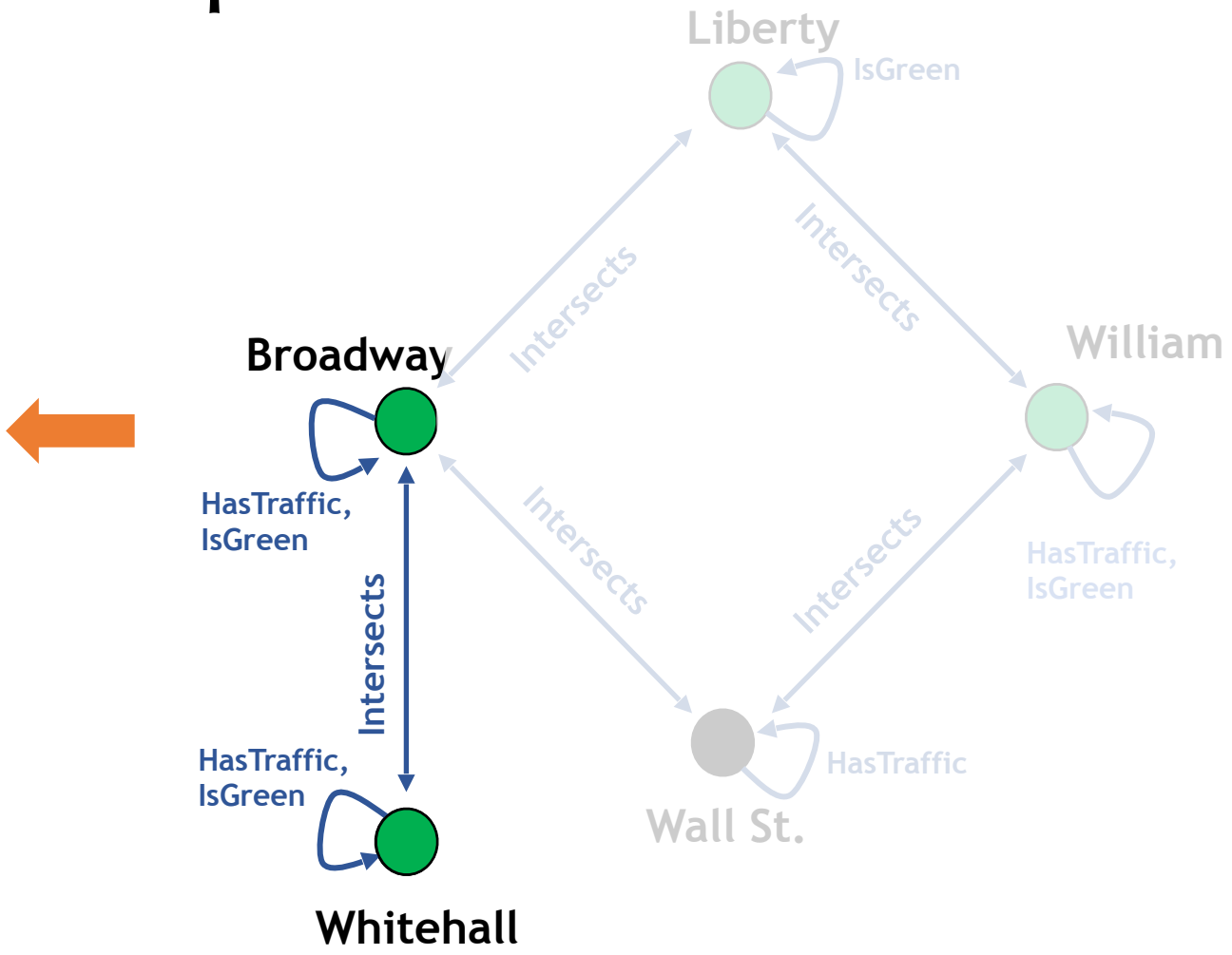


An Example

Crashes
Broadway
Whitehall St

Crashes(x) :– HasTraffic(x), isGreen(x),
Intersects(x, y),
HasTraffic(y), isGreen(y).

Crashes
Broadway
Whitehall St



Guarantees

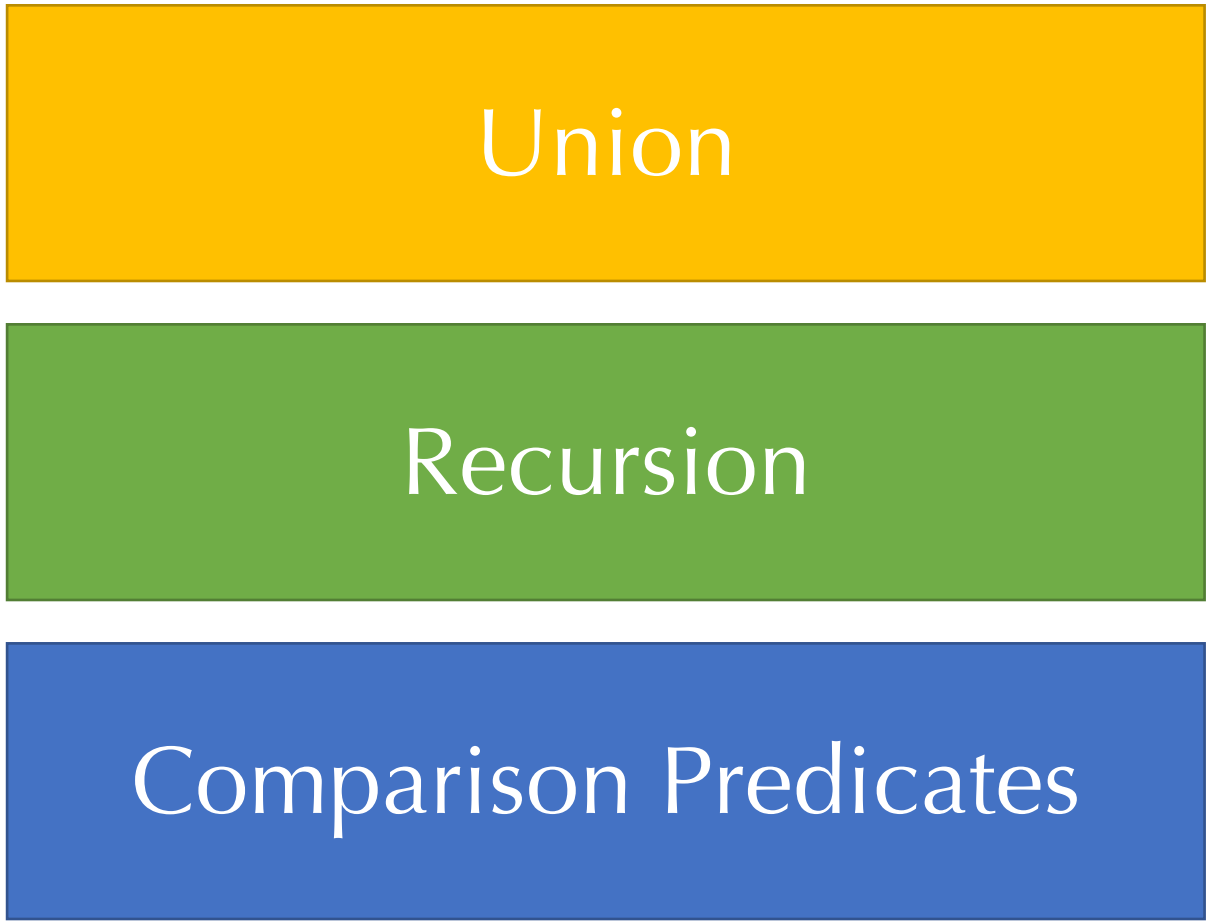
1. EGS is terminating as there are finitely many subgraphs.

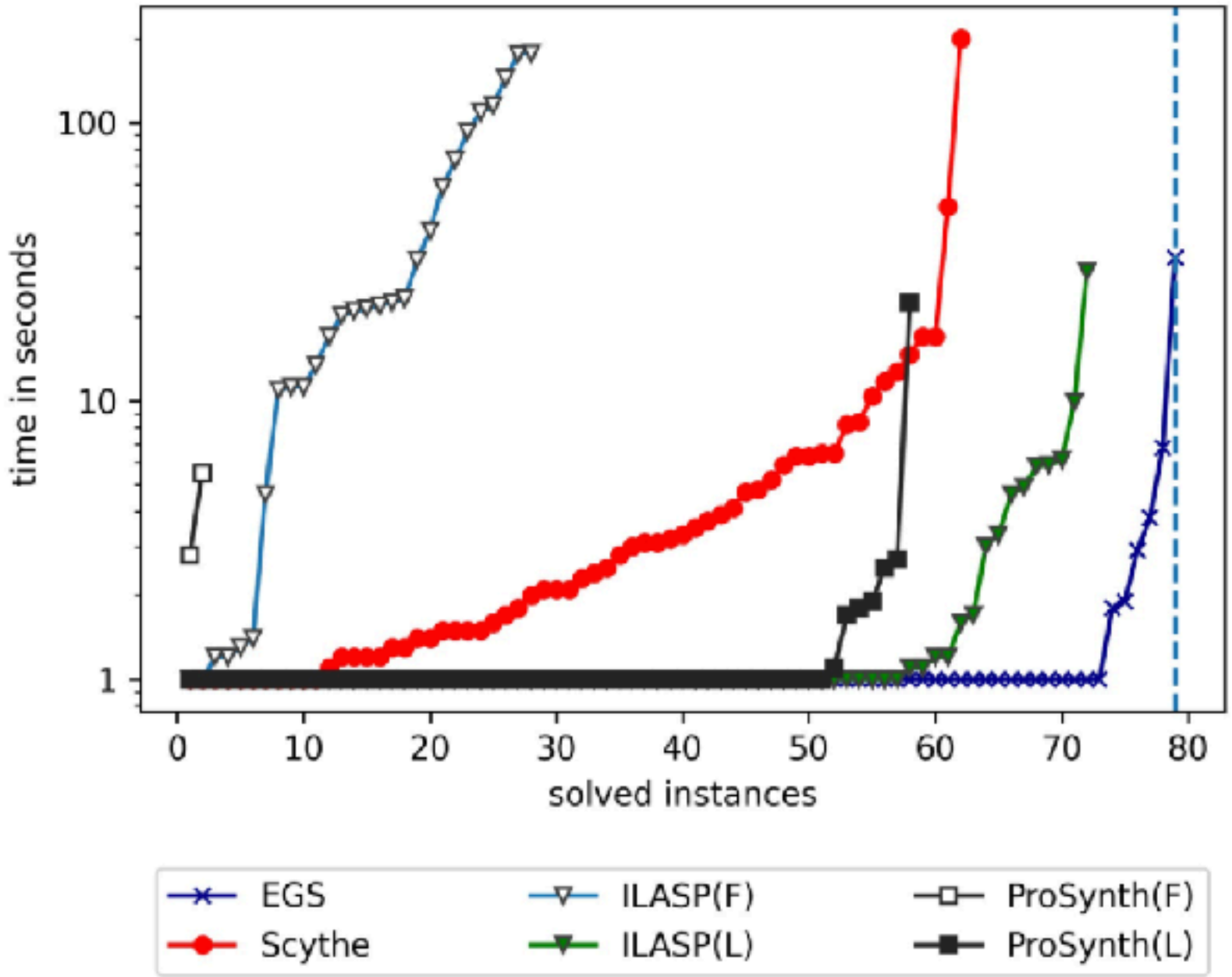
2. EGS is sound because consistency is verified as a part of synthesis.

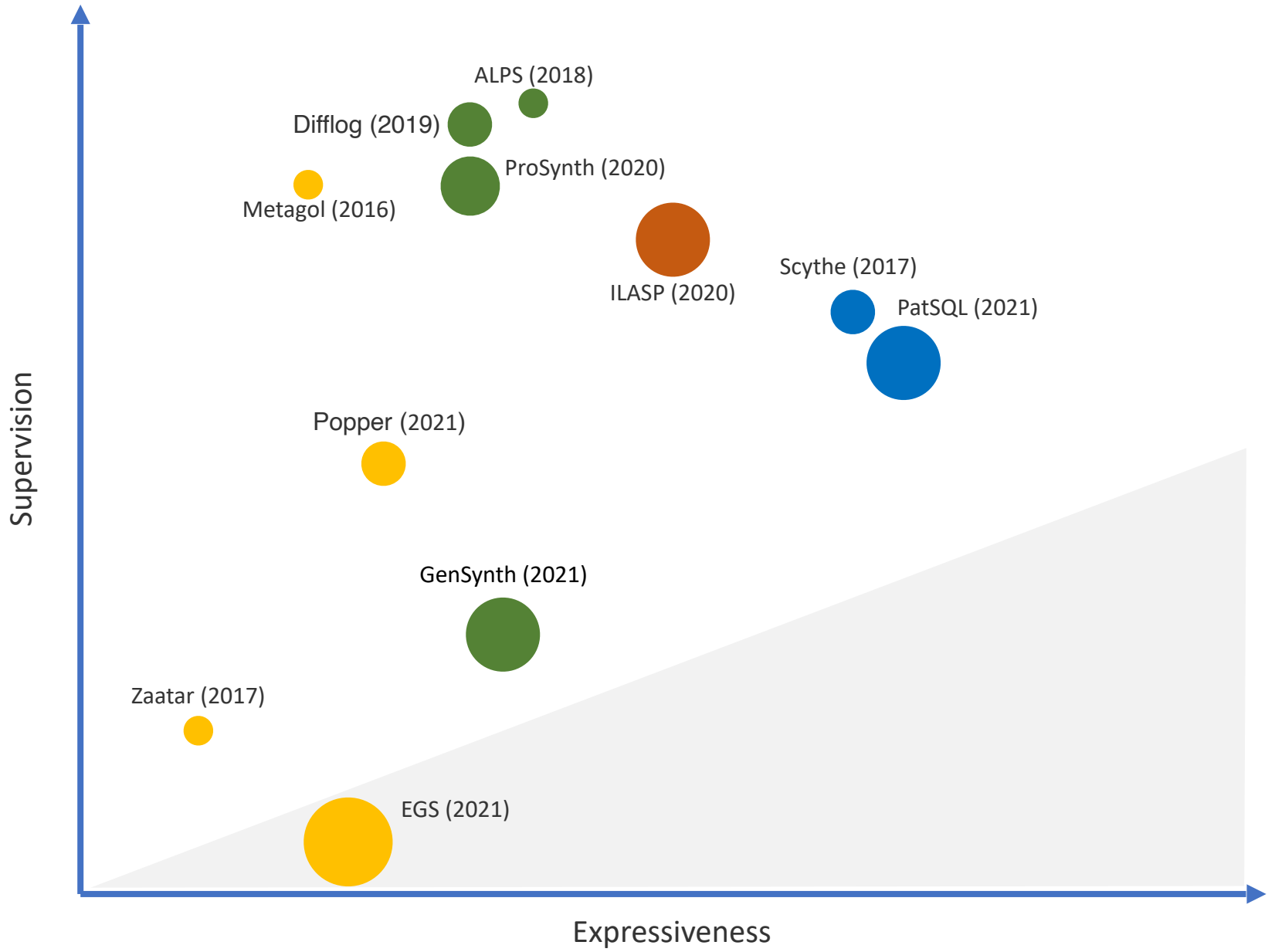
3. EGS is complete because:

   the query corresponding to the entire graph is consistent with the examples if and only if some consistent query exists.

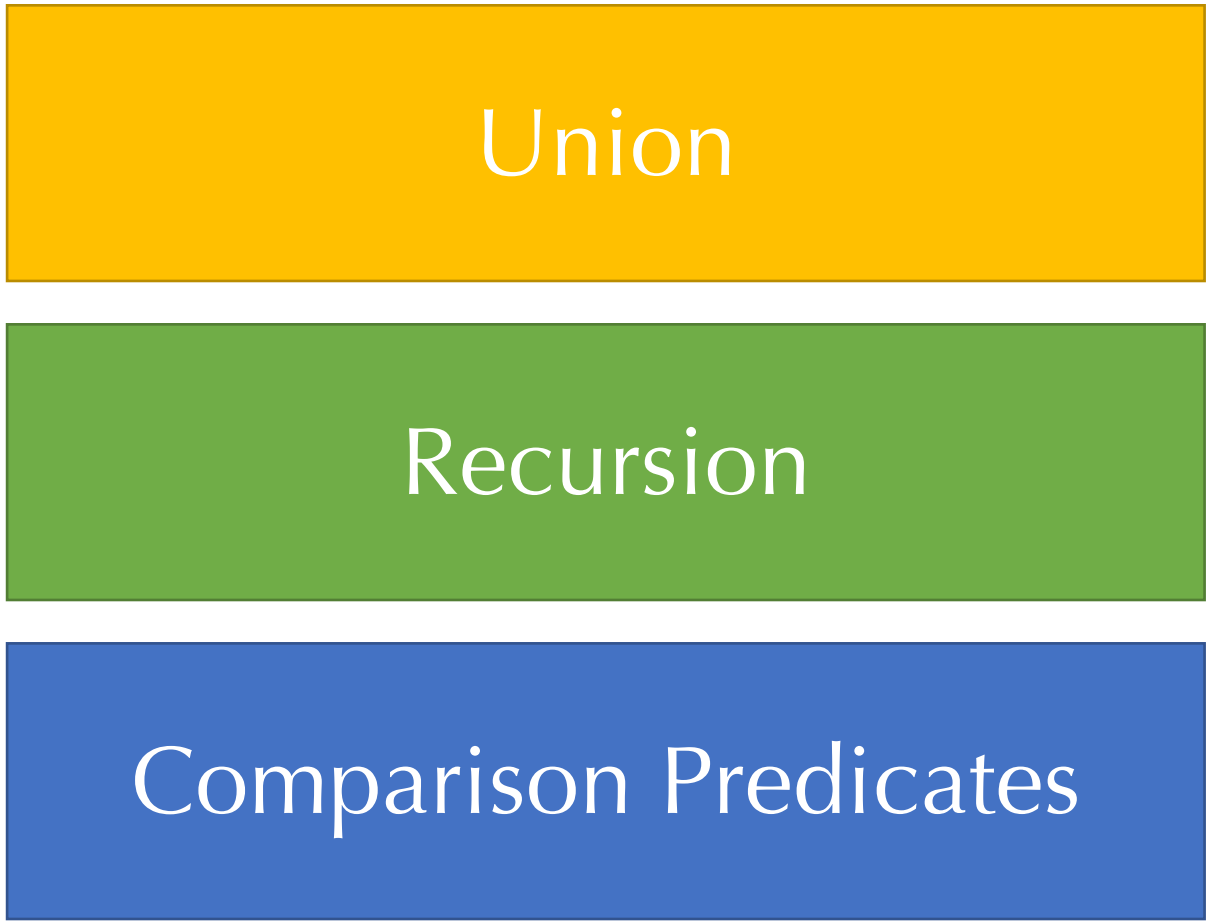
Extensions

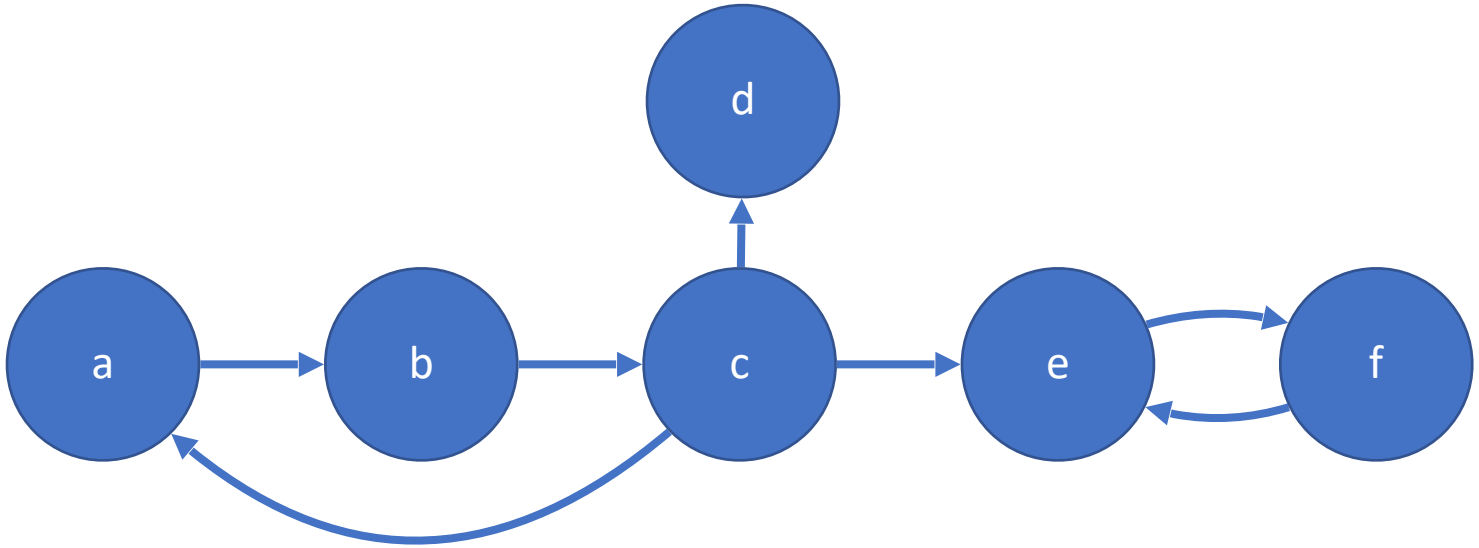



100
10
1
time in seconds
0
10
20
30
40
50
60
70
80
solved instances
EGS
Scythe
ILASP(F)
ILASP(L)
ProSynth(F)
ProSynth(L)

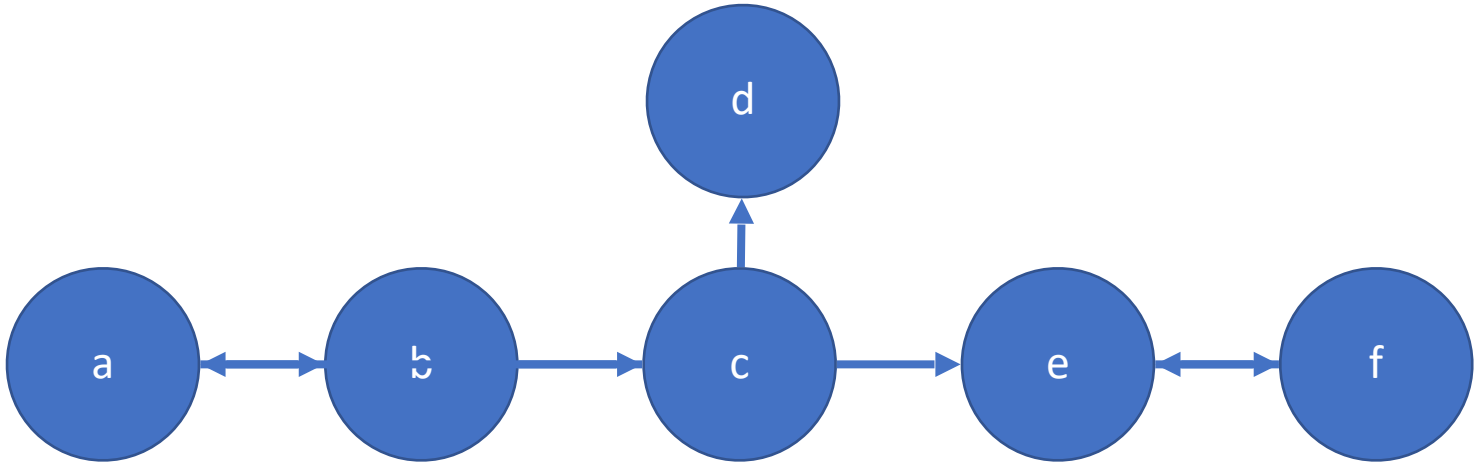

ALPS (2018)
Difflog (2019)
ProSynth (2020)
Metagol (2016)
ILASP (2020)
Scythe (2017)
PatSQL (2021)
Popper (2021)
GenSynth (2021)
Zaatar (2017)
EGS (2021)
Supervision
Expressiveness

Extensions

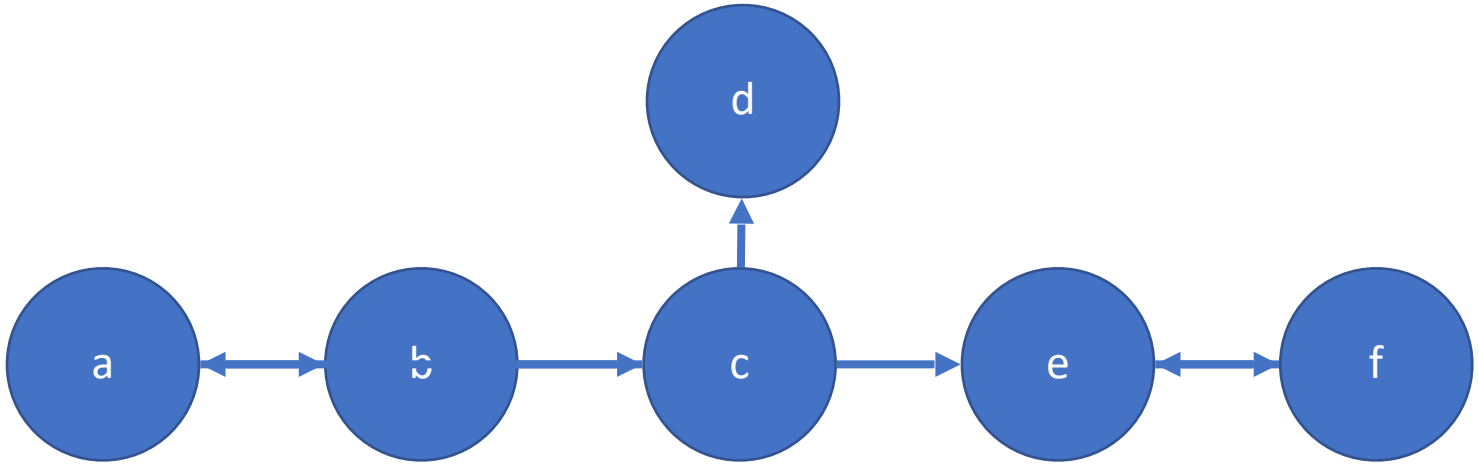




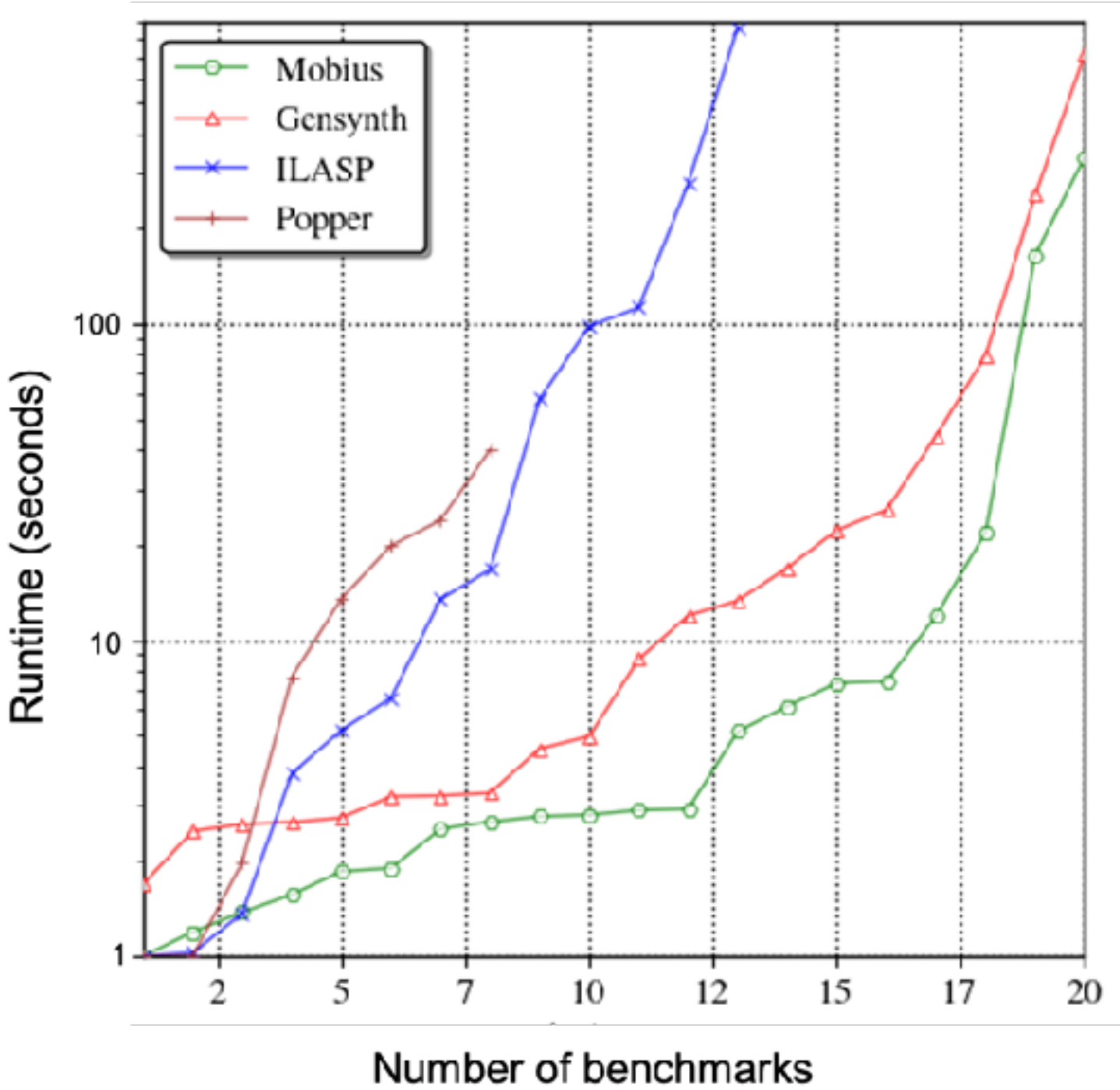

```
scc(x, y)  :–  path(x, y), path(y, x).
path(x, y)  :–  edge(x, y).
path(x, y)  :–  path(x, z), path(z,y).
```

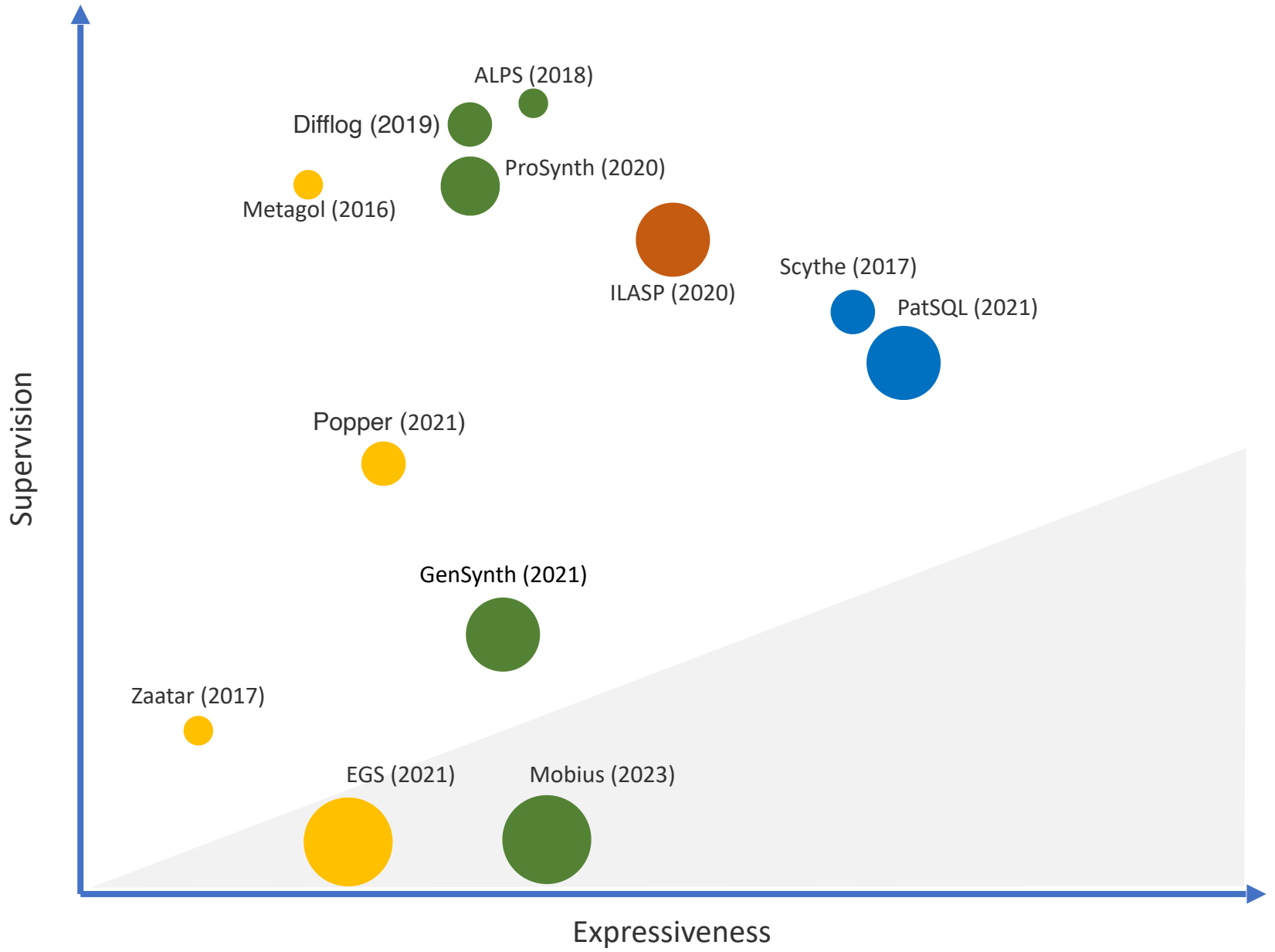



```
scc(x, y)  : –  path(x, y), path(y, x).
path(x, y)  : –  edge(x, y).
path(x, y)  : –  path(x, z), path(z,y).
```

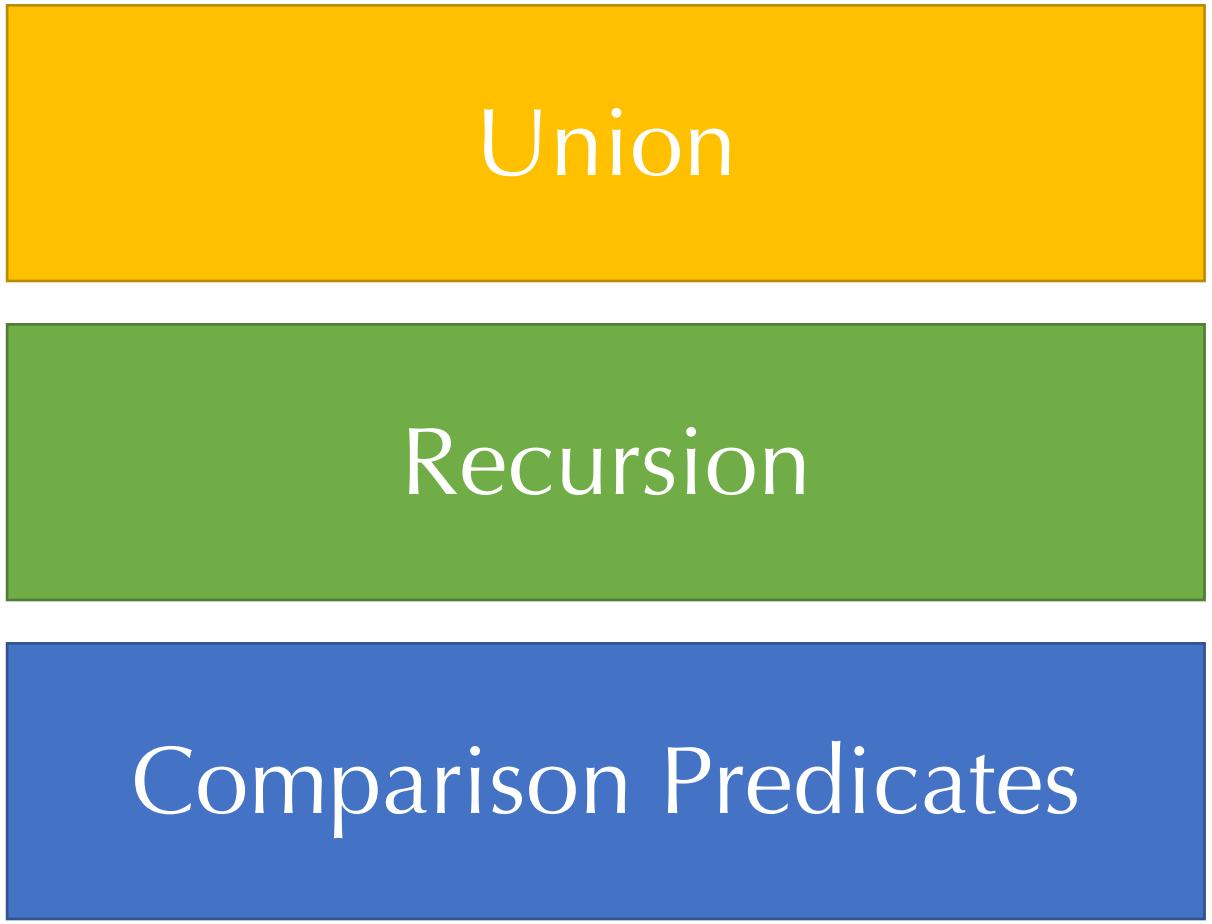



```
scc(x, y)  :– path(x, y), path(y, x).
path(x, y) :– edge(x, y).
path(x, y) :– path(x, z), path(z,y).
```


Mobius
Gensynth
ILASP
Popper
100
10
1
Runtime (seconds)
2
5
7
10
12
15
17
20
Number of benchmarks


ALPS (2018)
Difflog (2019)
ProSynth (2020)
Metagol (2016)
ILASP (2020)
Scythe (2017)
PatSQL (2021)
Popper (2021)
GenSynth (2021)
Zaatar (2017)
EGS (2021)
Mobius (2023)
Supervision
Expressiveness

Extensions



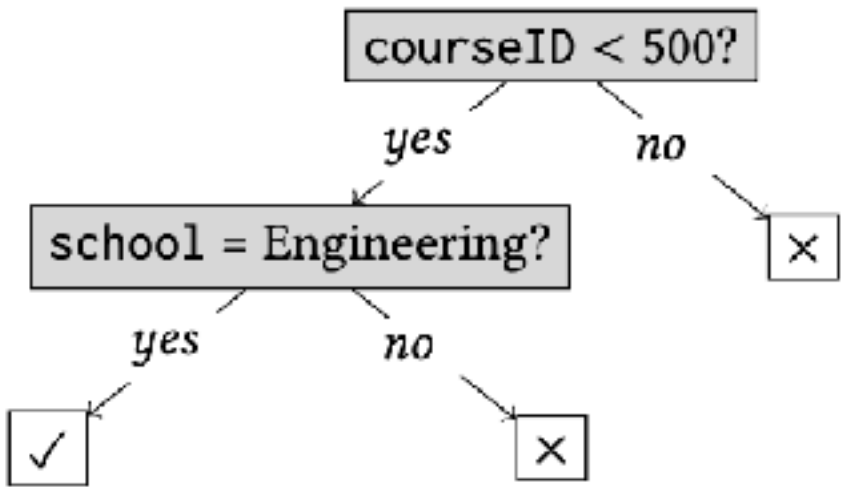
```
SELECT registration.studentID
    FROM registration JOIN department
        ON registration.deptCode = department.deptCode
    WHERE registration.courseID < 500
        AND department.school = "Engineering"
```

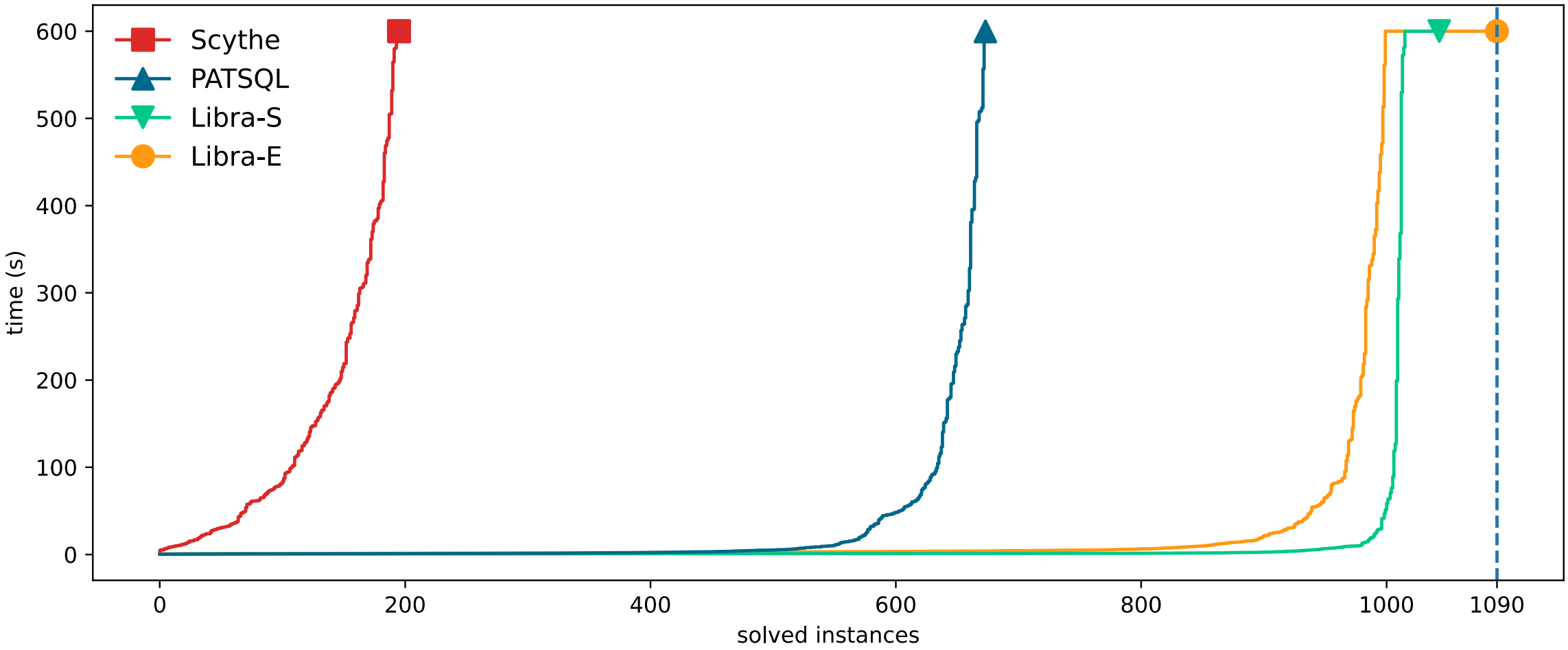
```
SELECT registration.studentID
    FROM registration JOIN department
        ON registration.deptCode = department.deptCode
```

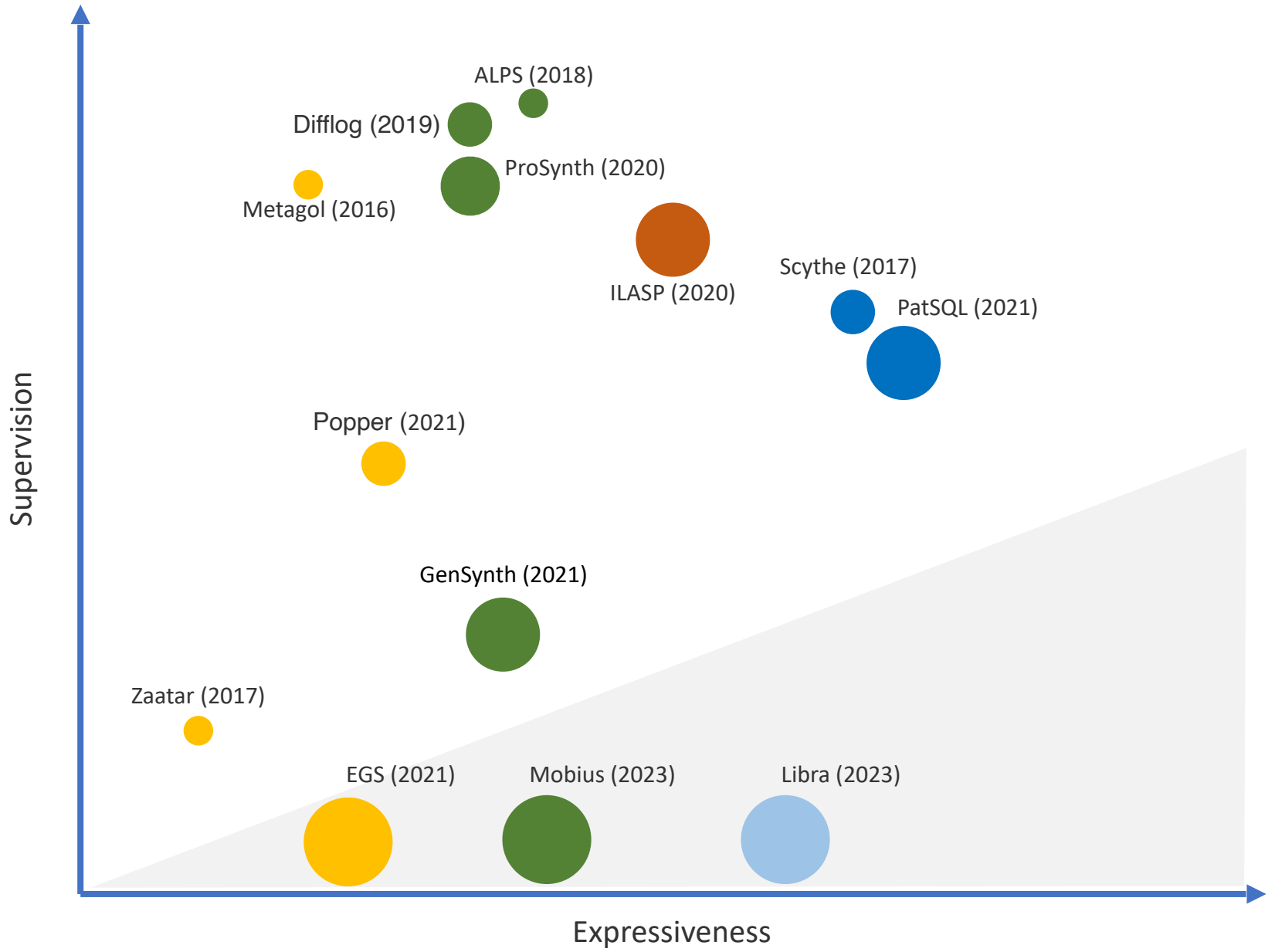
```
SELECT registration.studentID
    FROM registration JOIN department
        ON registration.deptCode = department.deptCode
    WHERE registration.courseID < 500
        AND department.school = "Engineering"
```

```
SELECT registration.studentID
    FROM registration JOIN department
        ON registration.deptCode = department.deptCode
    WHERE registration.courseID < 500
        AND department.school = "Engineering"
```

studentID	deptCode	courseID	school
Alice	Comp.	201	Engineering
Alice	Chem.	310	Arts and Science
Alice	Mech.	550	Engineering
Bob	Mech.	320	Engineering
Bob	Mech.	550	Engineering
Charlie	Chem.	310	Arts and Science
David	Comp.	500	Engineering
David	Mech.	502	Engineering
Erin	Chem.	310	Arts and Science




Scythe
PATSQL
Libra-S
Libra-E
time (s)
0
100
200
300
400
500
600
0
200
400
600
800
1000
1090
solved instances

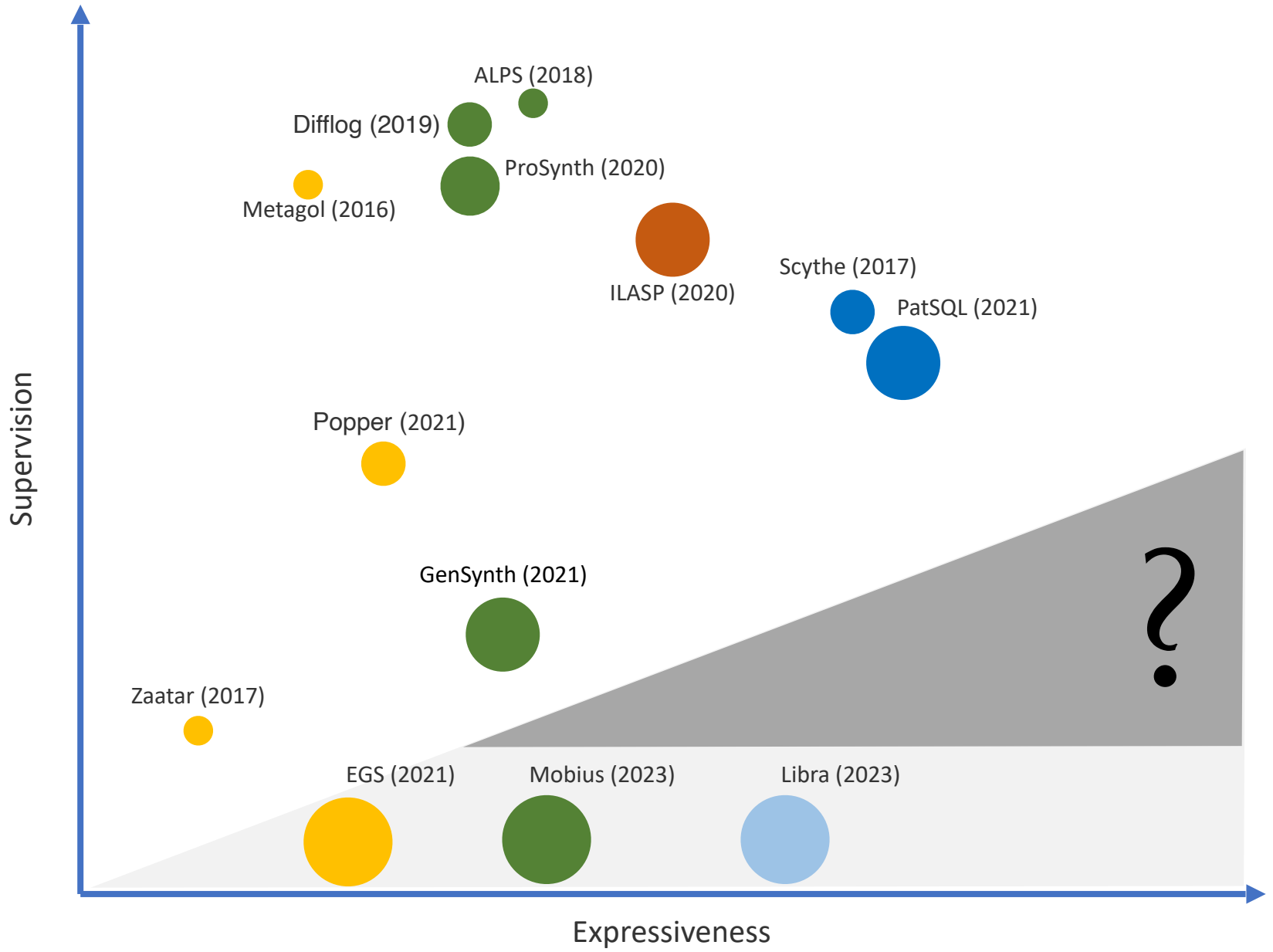

ALPS (2018)
Difflog (2019)
ProSynth (2020)
Metagol (2016)
ILASP (2020)
Scythe (2017)
PatSQL (2021)
Popper (2021)
GenSynth (2021)
Zaatar (2017)
EGS (2021)
Mobius (2023)
Libra (2023)
Supervision
Expressiveness

Future Directions

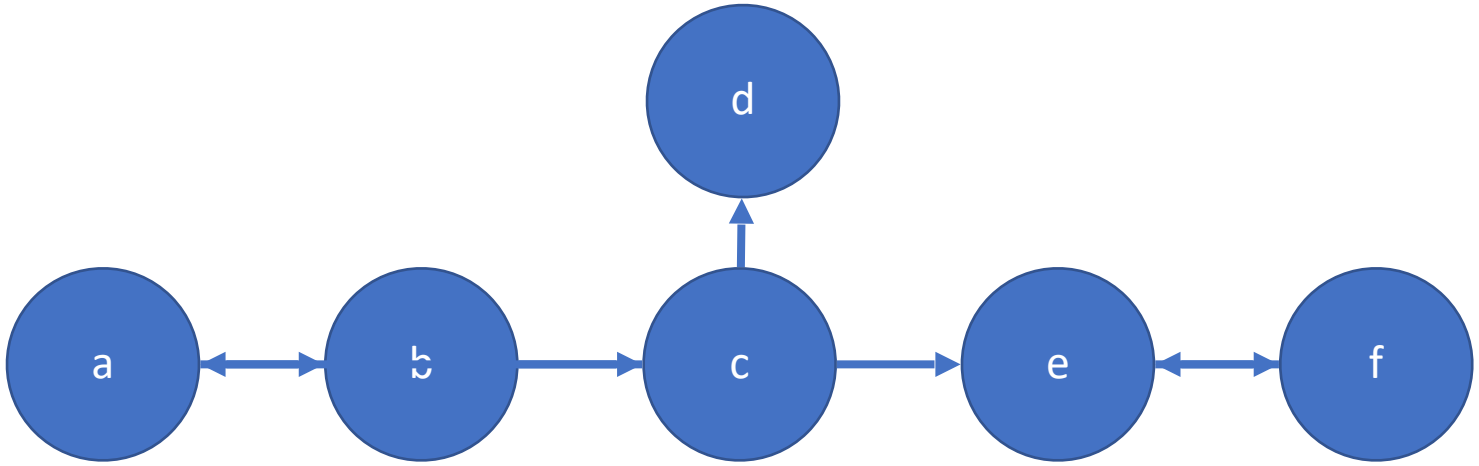
Synthesis in Presence of Noise

Scalability

More Expressibility (Aggregation)


ALPS (2018)
Difflog (2019)
ProSynth (2020)
Metagol (2016)
ILASP (2020)
Scythe (2017)
PatSQL (2021)
Popper (2021)
GenSynth (2021)
Zaatar (2017)
EGS (2021)
Mobius (2023)
Libra (2023)
?
Supervision
Expressiveness


Pre-doctoral Research Workshop
January 9-11, 2025
tinyurl.com/ashokacs-prw2025




```
scc(x, y) :- edge(x, y), edge(y, x).
scc(x, y) :- edge(x, y), edge(y, z), edge(z, x).
scc(x, z) :- edge(x, y), edge(y, z), edge(z, x).
```

Normalization

```
scc(x, y) :- edge(x, y), edge(y, x).
scc(x, y) :- edge(x, y), edge(y, z), edge(z, x).
scc(x, z) :- edge(x, y), edge(y, z), edge(z, x).
```

Normalization

scc(x, y) :– R(x, y), R(y, x).

scc(x, y) :– R(x, y), R(y, z), R(z, x).

scc(x, z) :– R(x, y), R(y, z), R(z, x).

R(x, y) :– edge(x, y).

Normalization

scc(x, y) :– R(x, y), R(y, x).

scc(x, y) :– R(x, y), S(y, x).

scc(x, z) :– S(x, z), R(z, x).

R(x, y) :– edge(x, y).

S(x, z) :– R(x, y), R(y, z).

Normalization

```
scc(x, y)  :- R(x, y), R(y, x).
scc(x, y)  :- S(x, z), R(z, x).
scc(x, z)  :- S(x, z), R(z, x).
R(x, y)    :- edge(x, y).
S(x, z)    :- R(x, y), R(y, z).
```

Unification

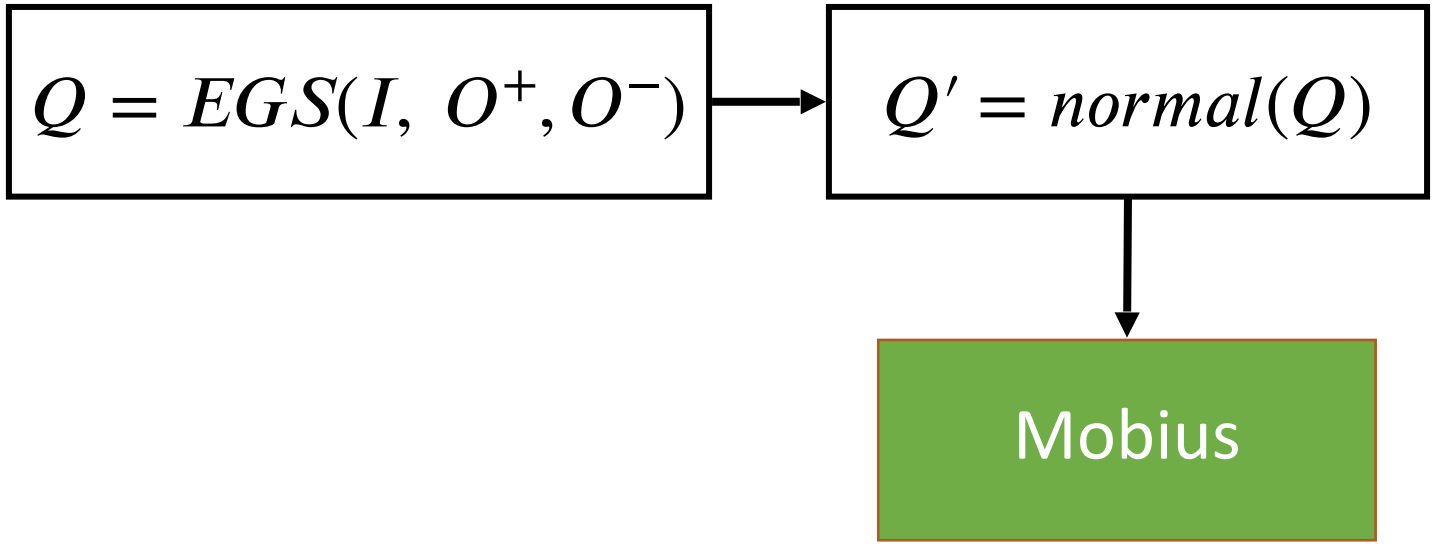
```
scc(x, y)  :- P(x, y), P(y, x).
scc(x, y)  :- P(x, y), P(z, y).
scc(x, z)  :- P(x, z), P(z, x).
P(x, y)    :- edge(x, y).
P(x, z)    :- P(x, y), P(y, z).
```

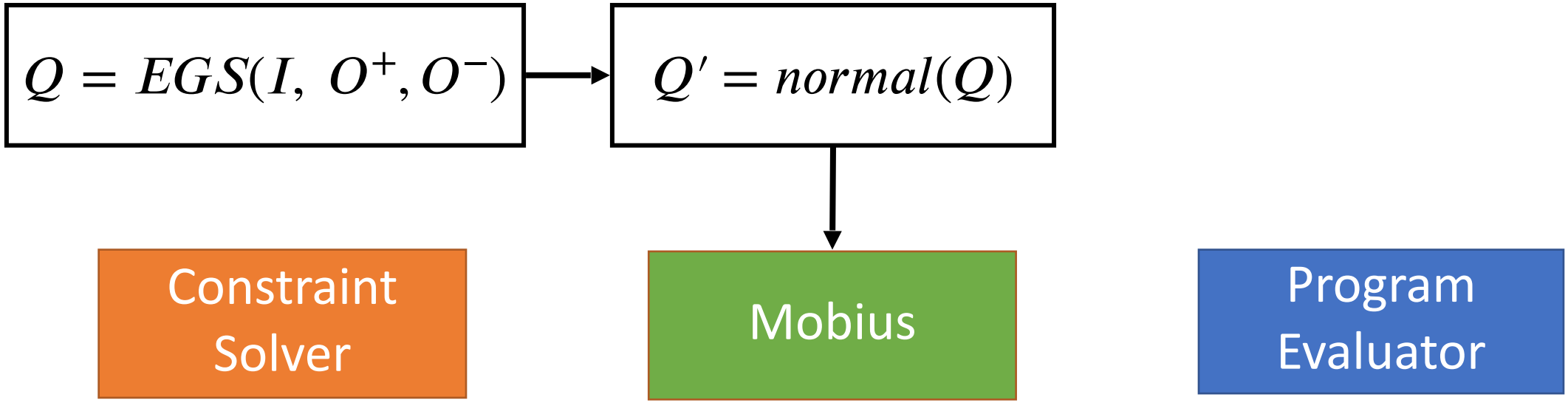
Unification

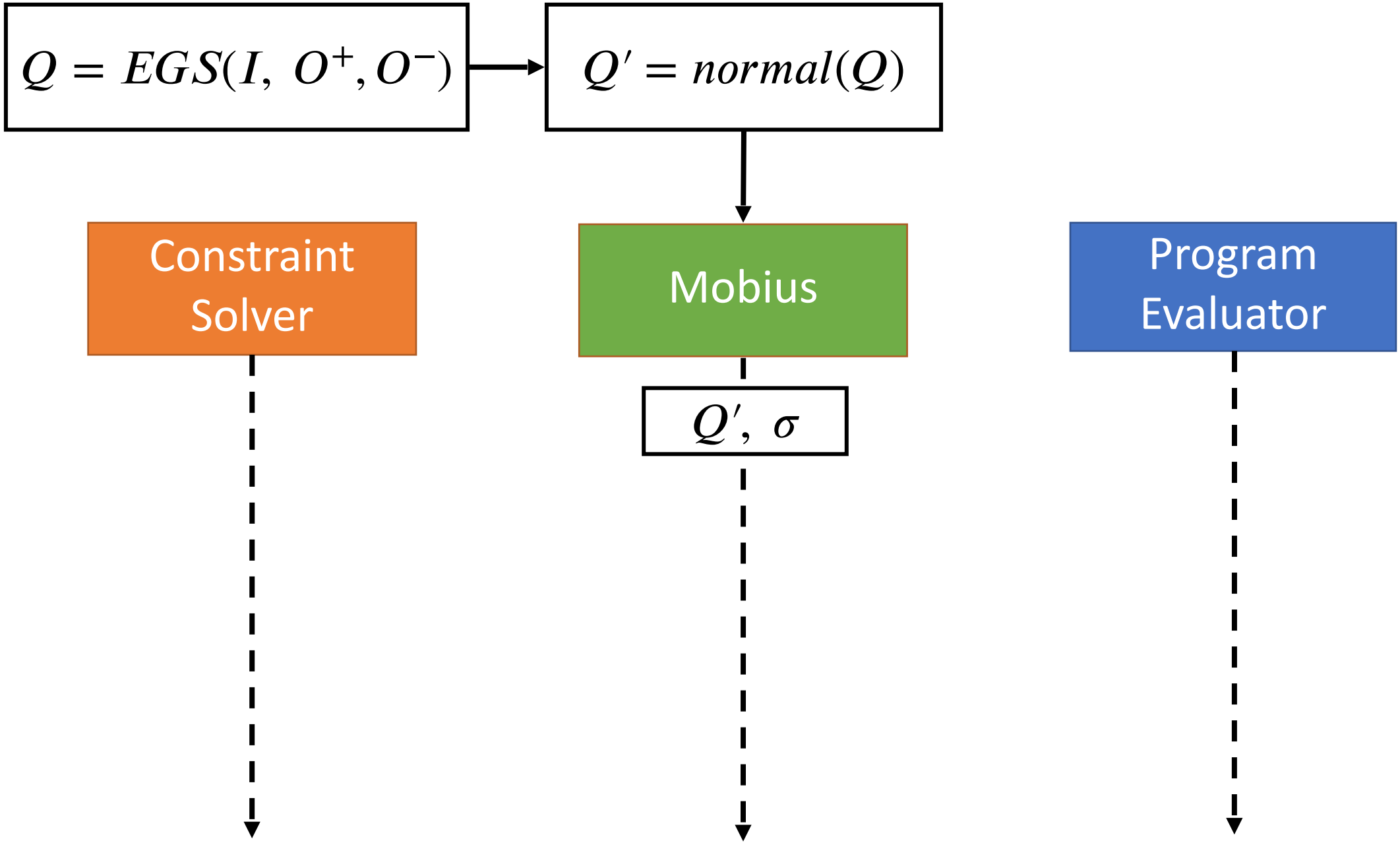
scc(x, y) :– P(x, y), P(y, x).

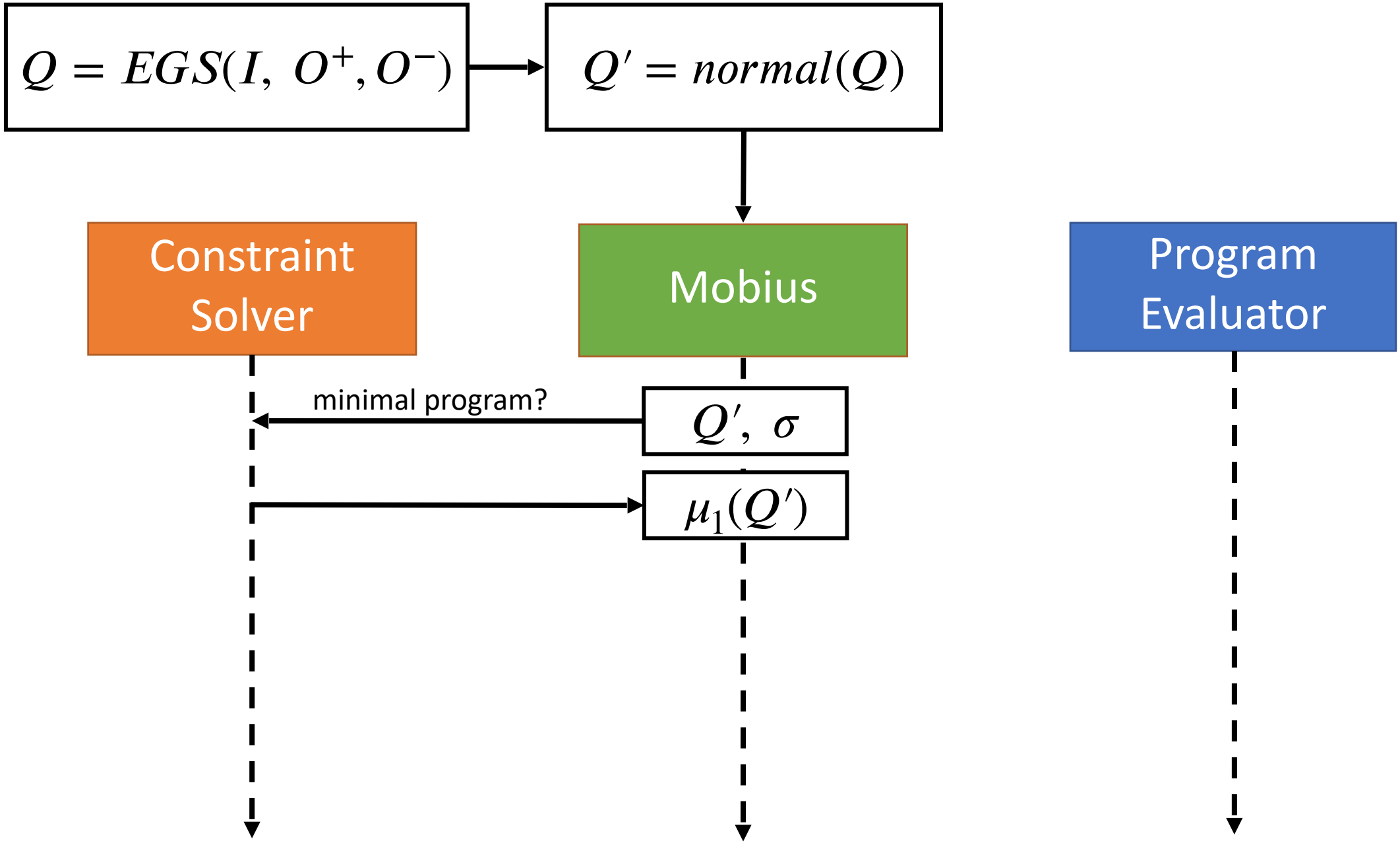
P(x, y) :– edge(x, y).

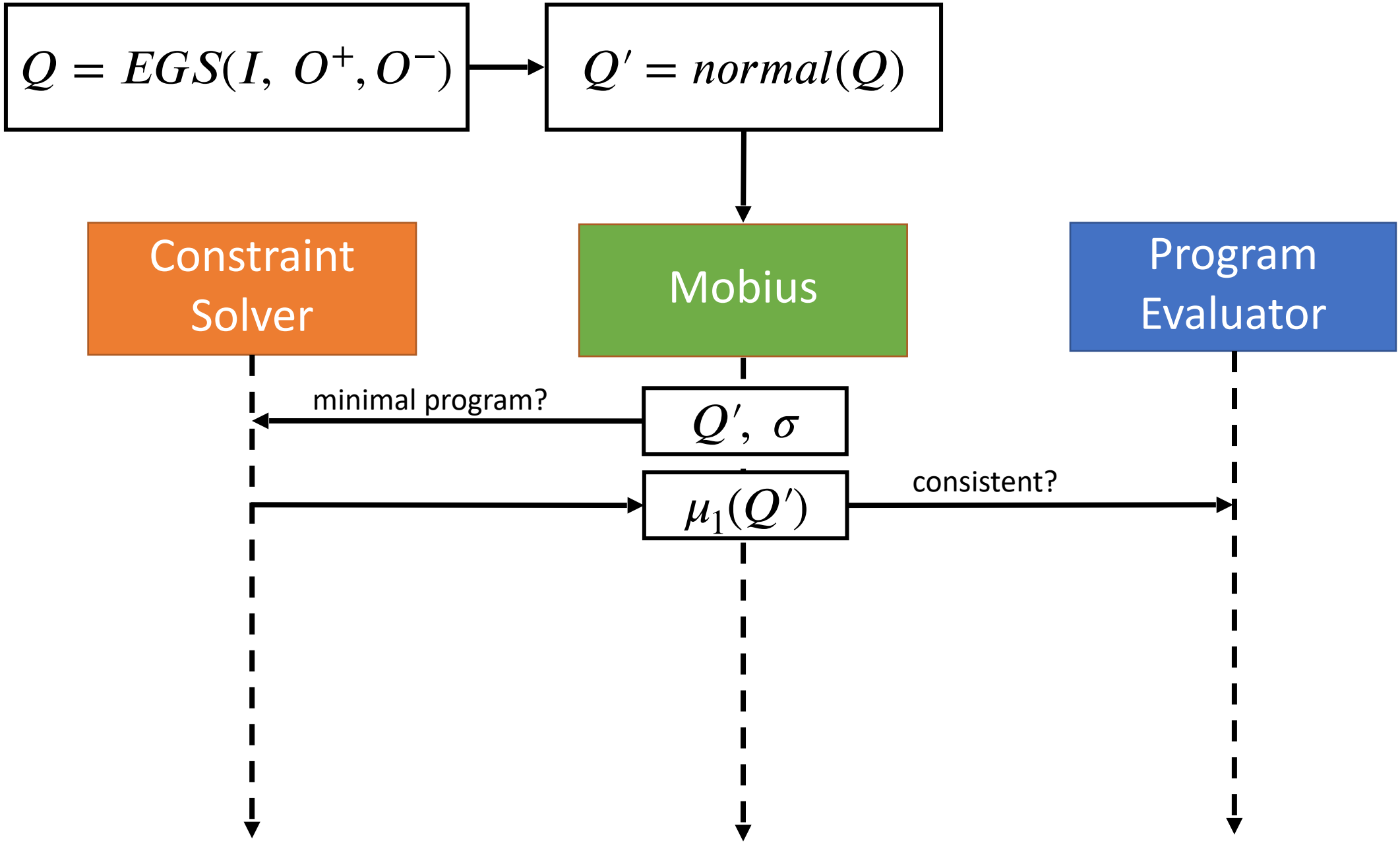
P(x, z) :– P(x, y), P(y, z).

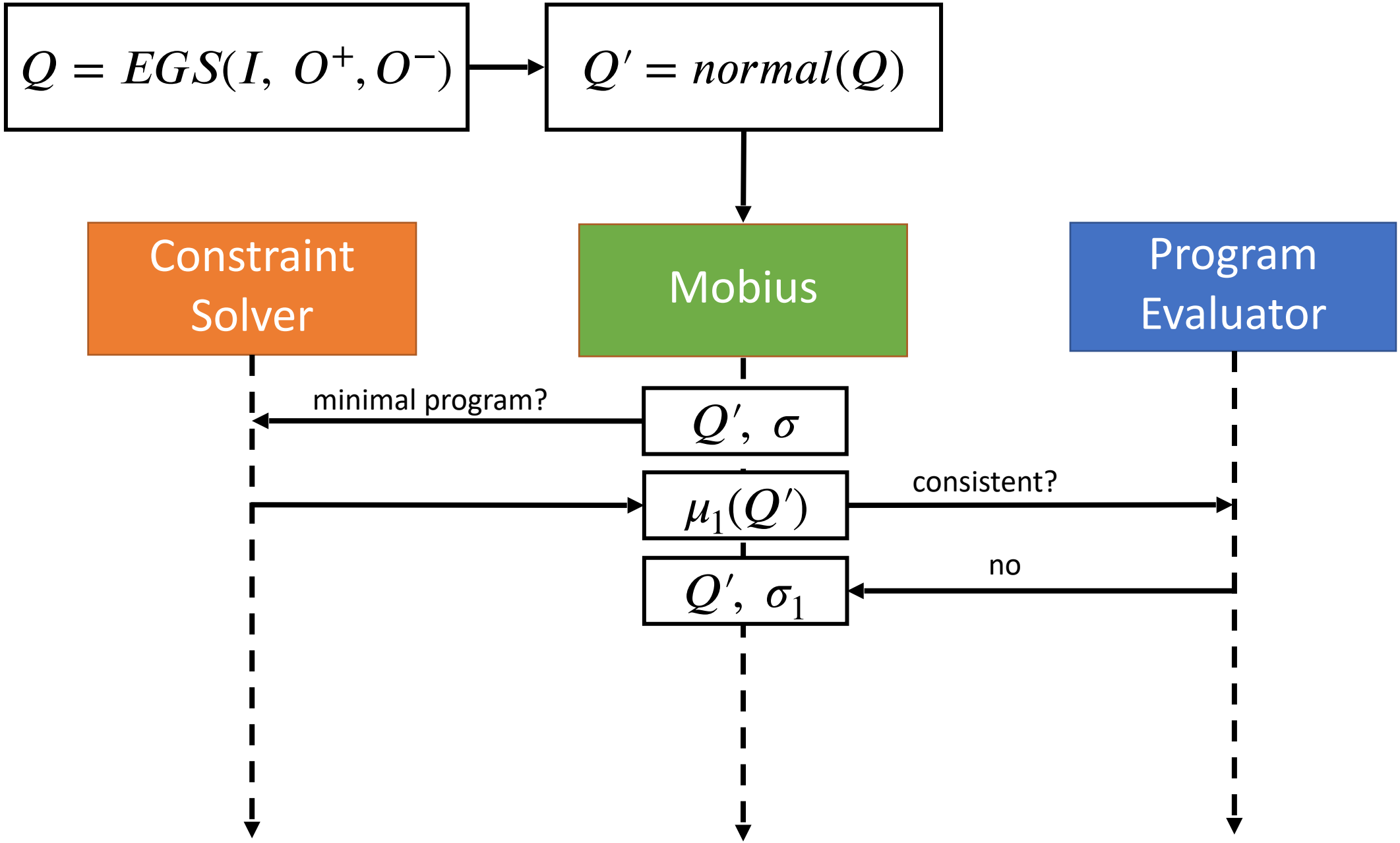

$Q = EGS(I,\ O^+, O^-)$
$Q' = normal(Q)$
Mobius

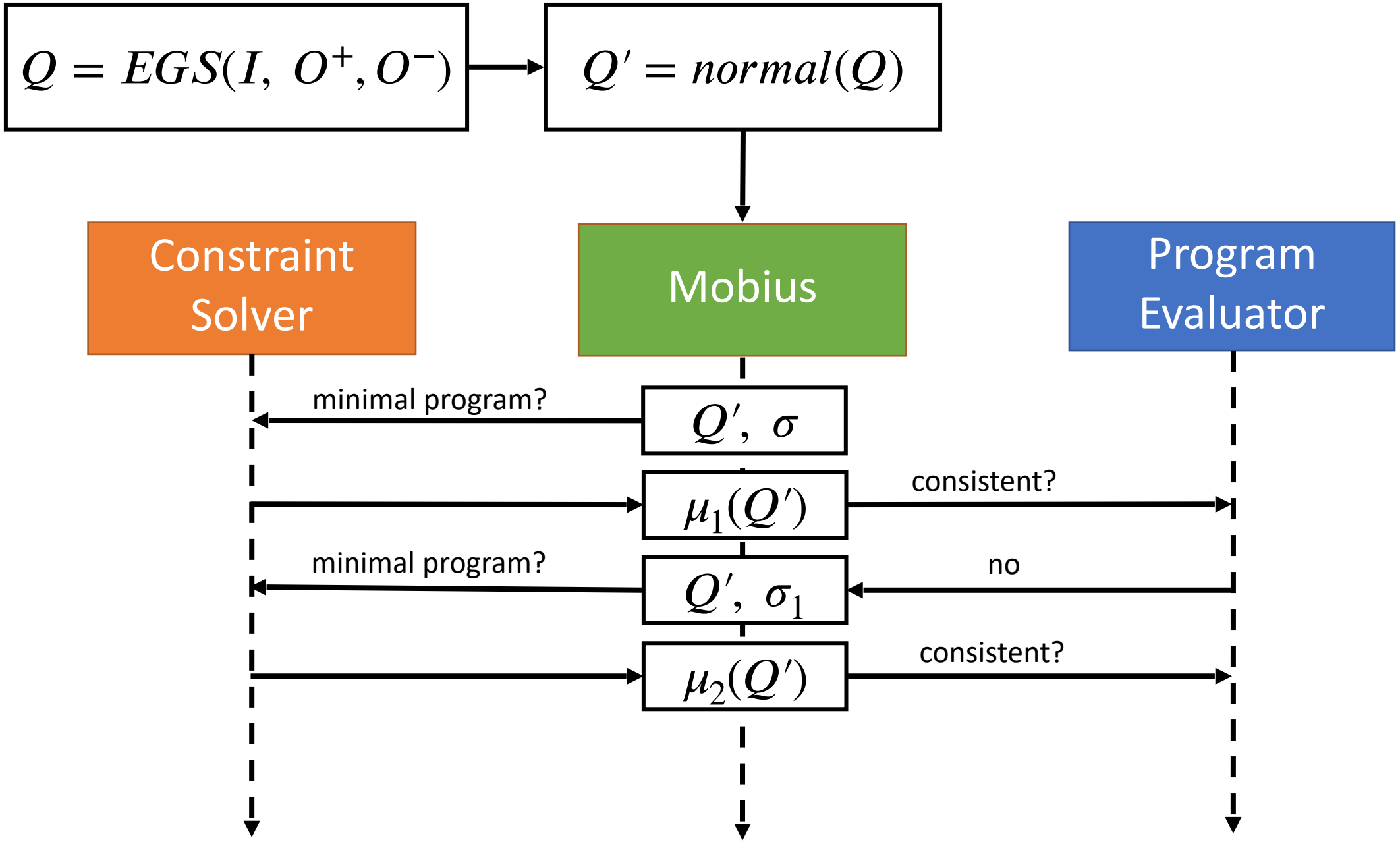

$Q = EGS(I,\ O^+, O^-)$
$Q' = normal(Q)$
Constraint Solver
Mobius
Program Evaluator

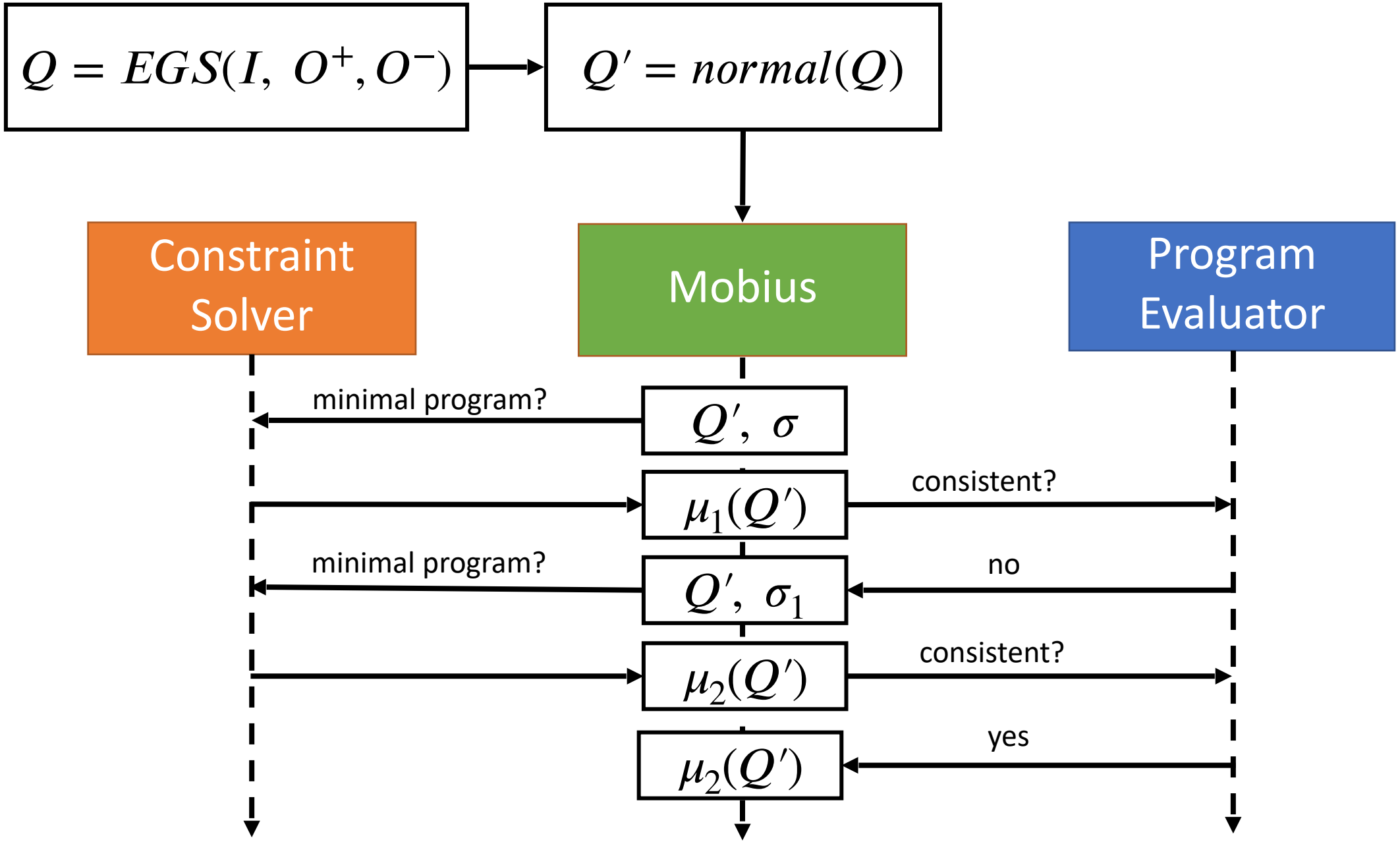

$Q = EGS(I, O^+, O^-)$
$Q' = normal(Q)$
Constraint Solver
Mobius
Program Evaluator
Q', σ

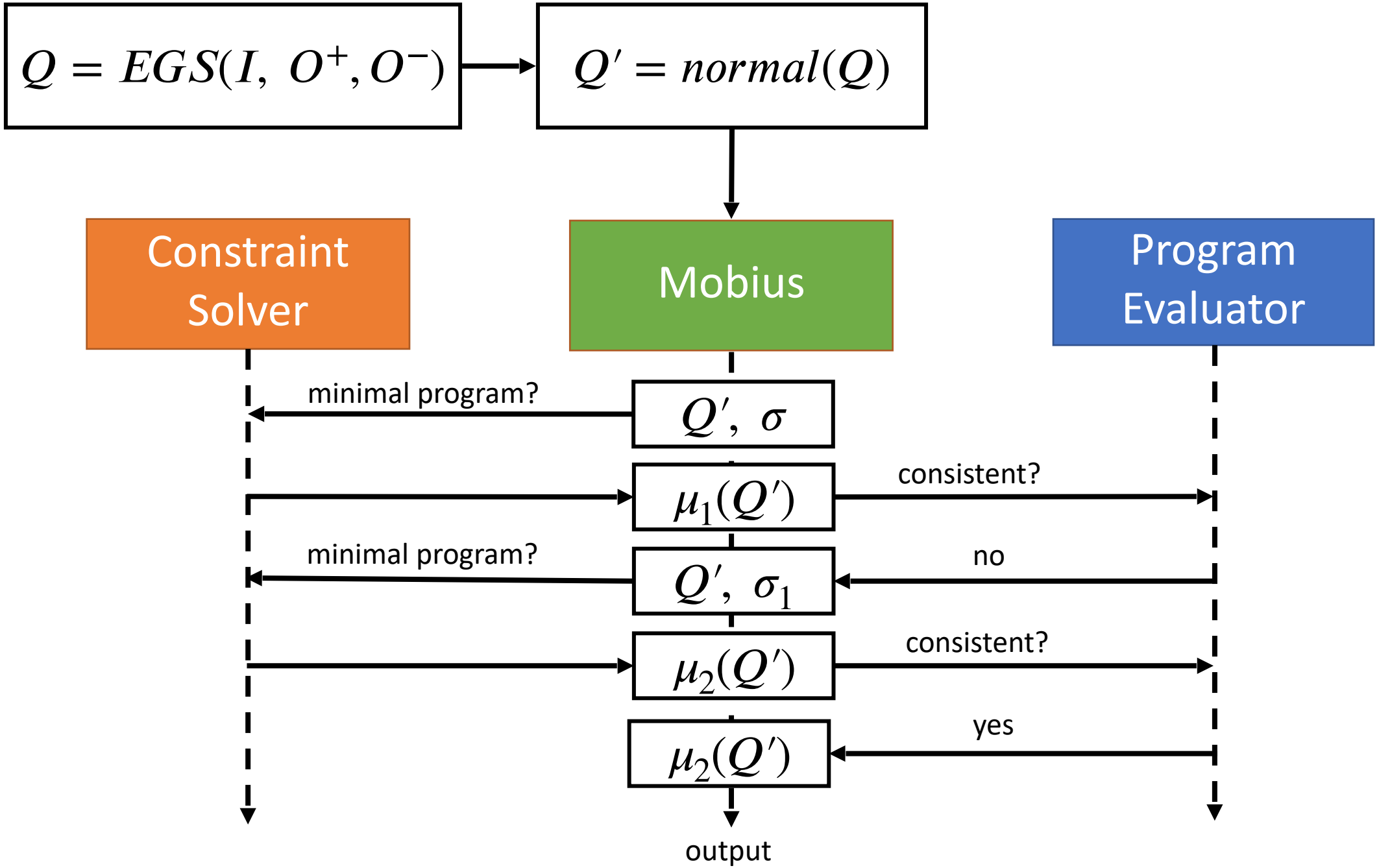

$Q = EGS(I, O^+, O^-)$
$Q' = normal(Q)$
Constraint Solver
Mobius
Program Evaluator
minimal program?
Q', σ
$\mu_1(Q')$


$Q = EGS(I, O^+, O^-)$
$Q' = normal(Q)$
Constraint Solver
Mobius
Program Evaluator
minimal program?
Q', σ
$\mu_1(Q')$
consistent?


$Q = EGS(I, O^+, O^-)$
$Q' = normal(Q)$
Constraint Solver
Mobius
Program Evaluator
minimal program?
Q', σ
$\mu_1(Q')$
consistent?
no
Q', σ_1


$Q = EGS(I, O^+, O^-)$
$Q' = normal(Q)$
Constraint Solver
Mobius
Program Evaluator
minimal program?
Q', σ
$\mu_1(Q')$
consistent?
minimal program?
Q', σ_1
no
$\mu_2(Q')$
consistent?


$Q = EGS(I, O^+, O^-)$
$Q' = normal(Q)$
Constraint Solver
Mobius
Program Evaluator
minimal program?
Q', σ
$\mu_1(Q')$
consistent?
minimal program?
Q', σ_1
no
$\mu_2(Q')$
consistent?
$\mu_2(Q')$
yes


$Q = EGS(I, O^+, O^-)$
$Q' = normal(Q)$
Constraint Solver
Mobius
Program Evaluator
minimal program?
Q', σ
$\mu_1(Q')$
consistent?
minimal program?
Q', σ_1
no
$\mu_2(Q')$
consistent?
$\mu_2(Q')$
yes
output