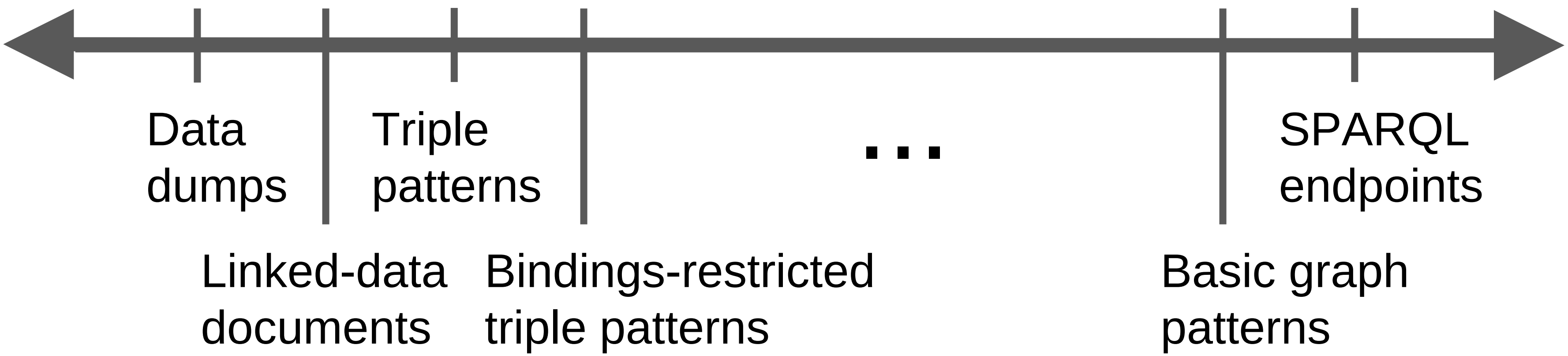
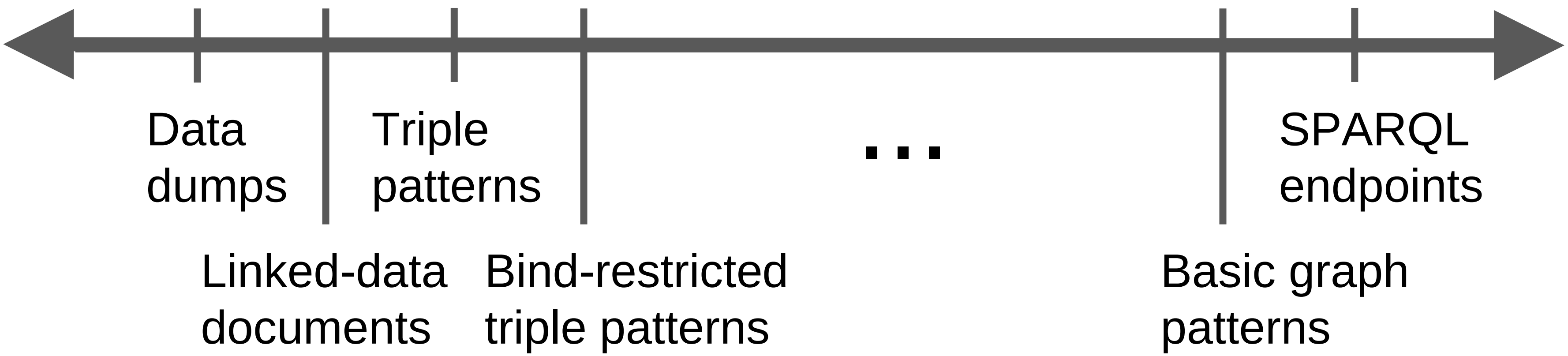
Linked Data Fragments World



Generic requests
High server availability
High client effort

Specific requests
Low server availability
High server effort

Is this really a line?



“Give me all the subjects and objects of triples whose predicate is rdf:type*”*

Is this really a line?

“Give me all persons reachable from Peter following two foaf:knows *links”*

A more fundamental understanding of LDF interfaces

Server	Can we analyze an interface before actually implementing it?
Client	What is the best way to use an interface given a specific budget?

A Formal Framework to Compare Linked Data Fragments

Olaf Hartig   Ian Letter   Jorge Pérez

Main contributions

Formal machine model for LDF settings

based on Turing Machines

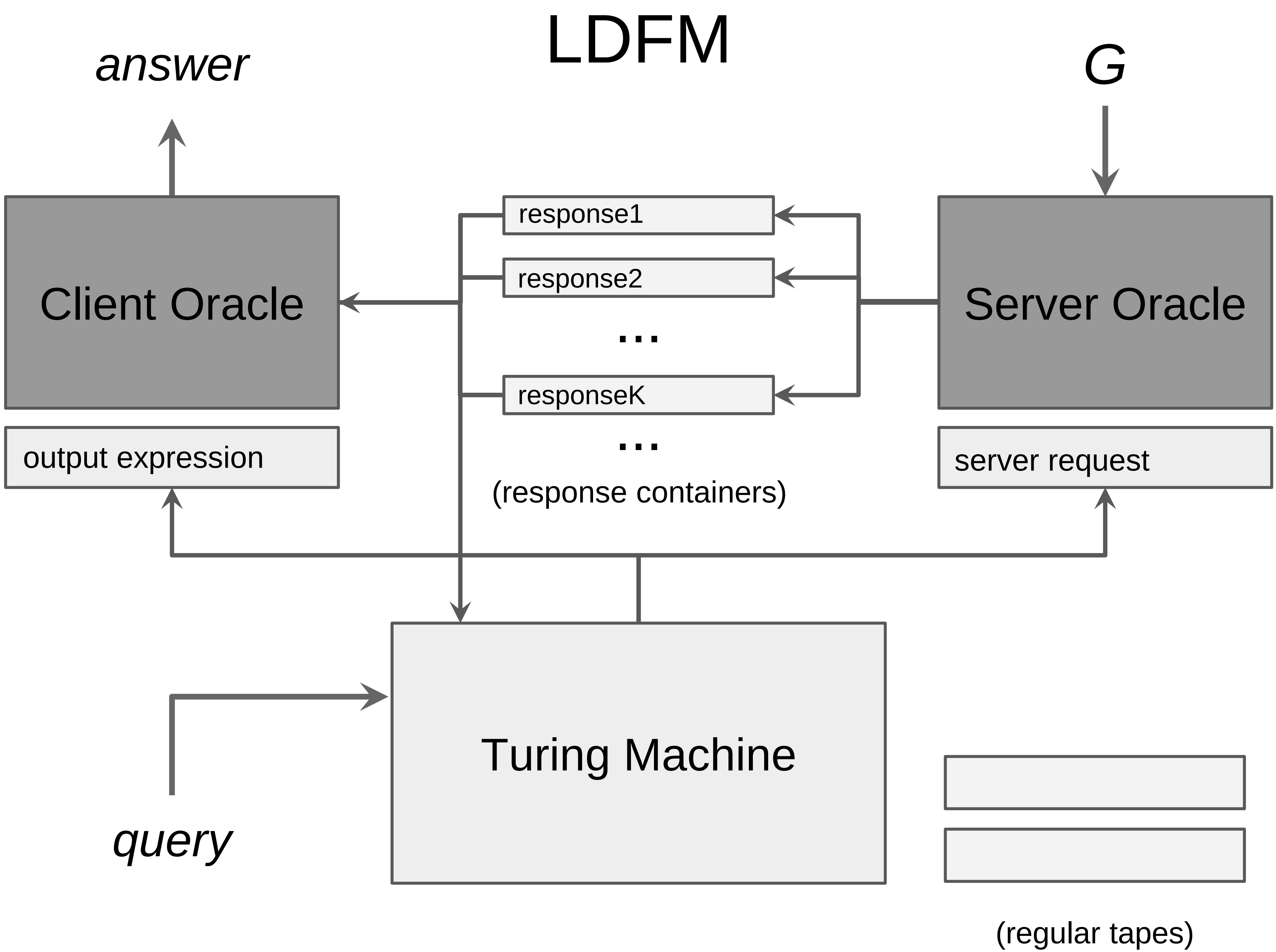
Complete expressiveness lattice

considering several combinations of interfaces

Fine grained complexity analysis

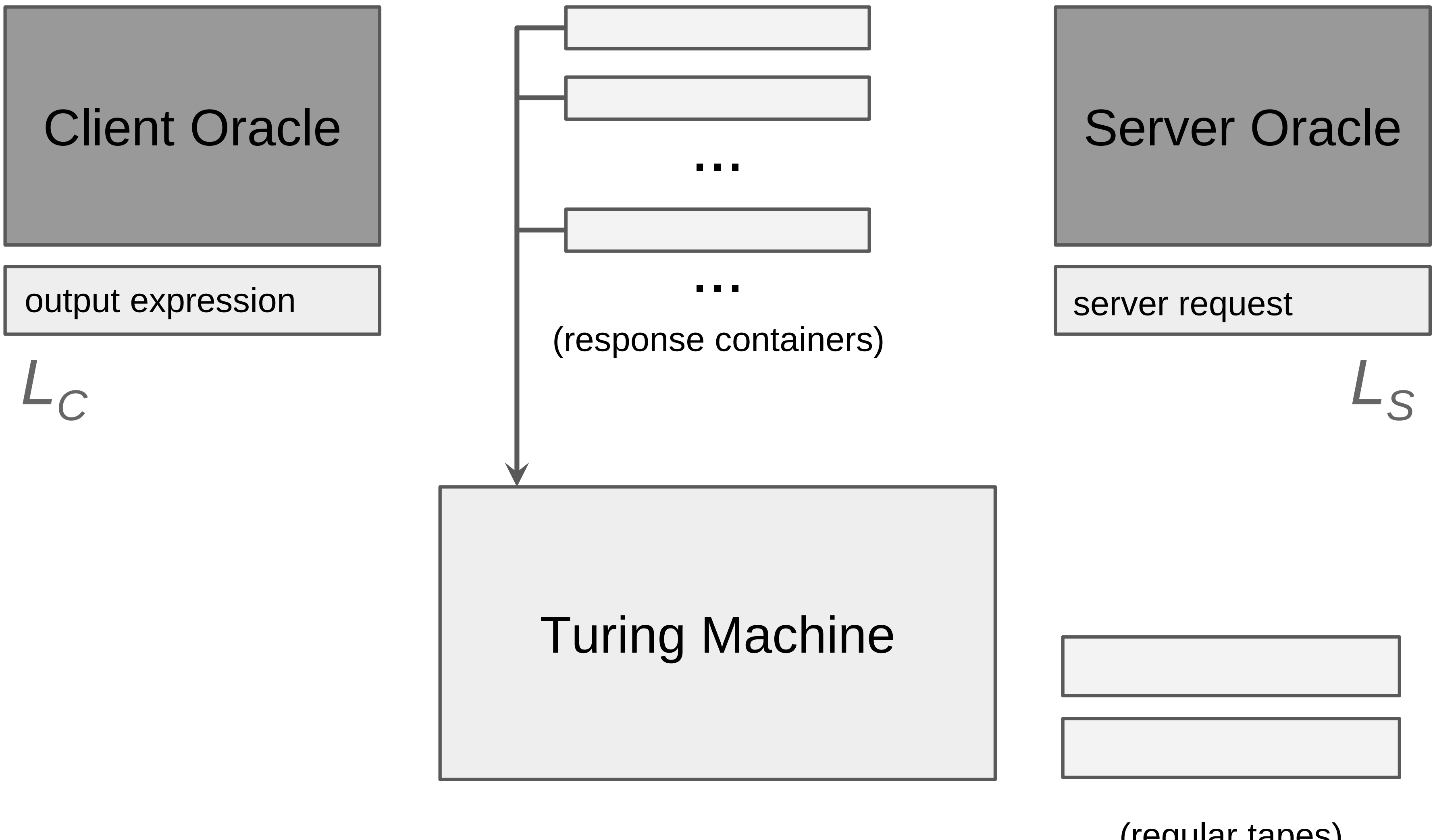
classical complexity, # requests, data transferred

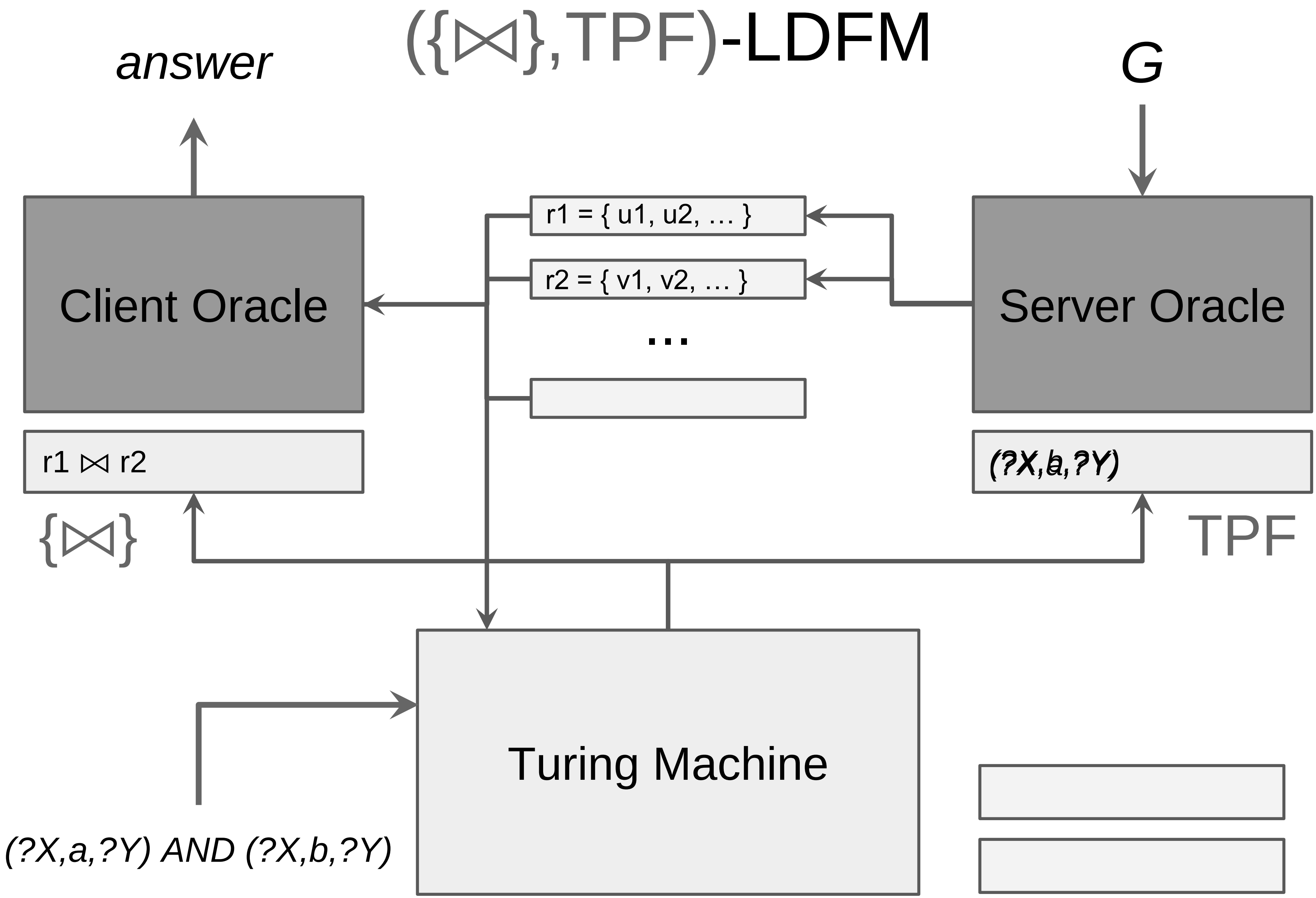
Linked Data Fragment Machine (LDFM)

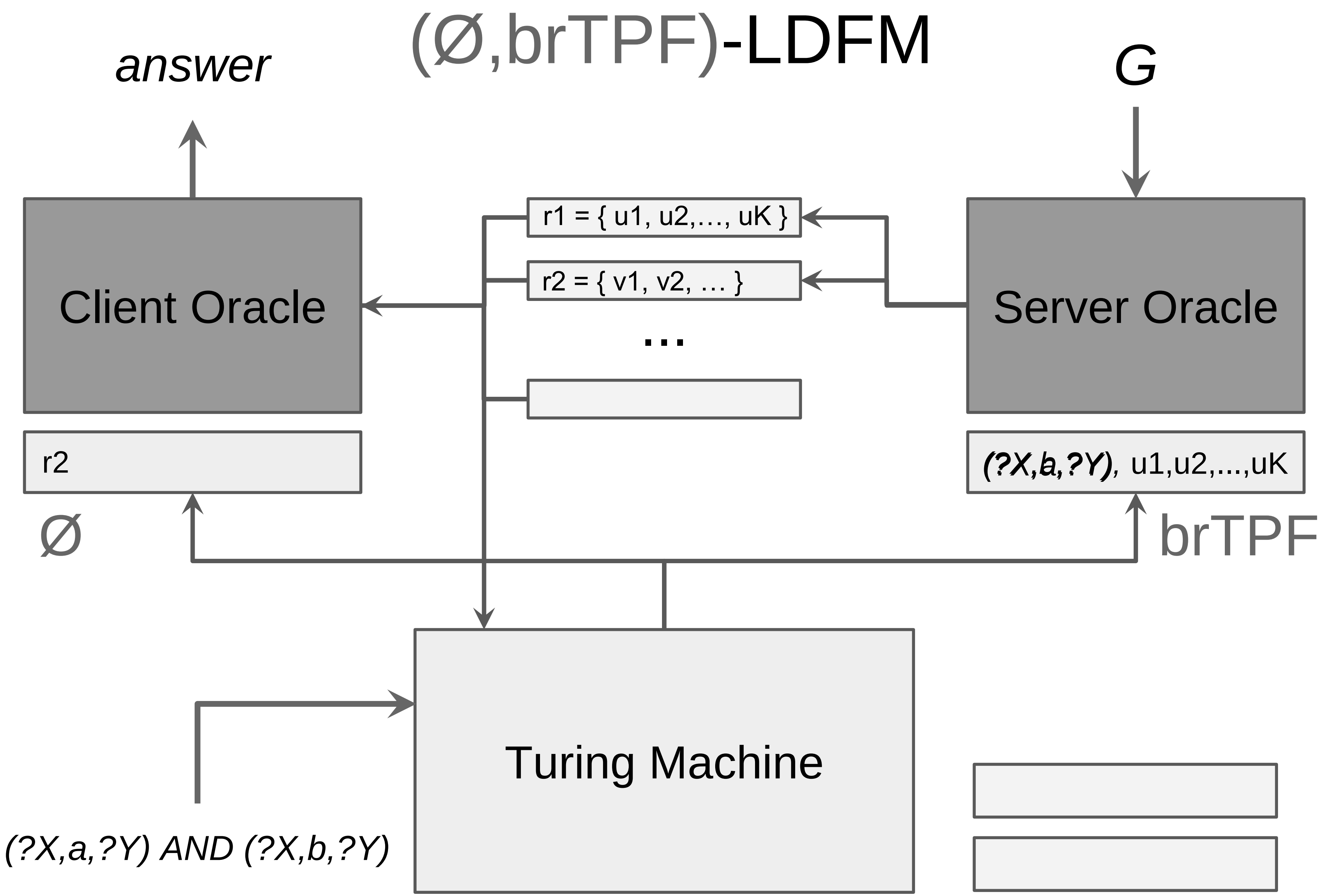


LDFM
answer
G
Client Oracle
response1
response2
...
responseK
...
(response containers)
Server Oracle
output expression
server request
Turing Machine
query
(regular tapes)

$(L_C L_S)$-LDFM


Client Oracle
output expression
L_C
...
...
(response containers)
Server Oracle
server request
L_S
Turing Machine
(regular tapes)


({⋈},TPF)-LDFM
answer
G
Client Oracle
Server Oracle
r1 = { u1, u2, … }
r2 = { v1, v2, … }
…
r1 ⋈ r2
(?X,a,?Y)
{⋈}
TPF
Turing Machine
(?X,a,?Y) AND (?X,b,?Y)



(Ø,brTPF)-LDFM
answer
G
Client Oracle
Server Oracle
r1 = { u1, u2,…, uK }
r2 = { v1, v2, … }
…
r2
(?X,b,?Y), u1,u2,...,uK
Ø
brTPF
Turing Machine
(?X,a,?Y) AND (?X,b,?Y)

What are the queries computed by $(L_C L_S)$ LDFMs?

$$(L_C L_S) \equiv (R_C R_S)$$

$$(L_C L_S) \prec (R_C R_S)$$

Expressiveness Lattice

Client Languages

subsets of

$\{ \cup, \bowtie, ⟕, \pi \}$

response combinations

Server Languages

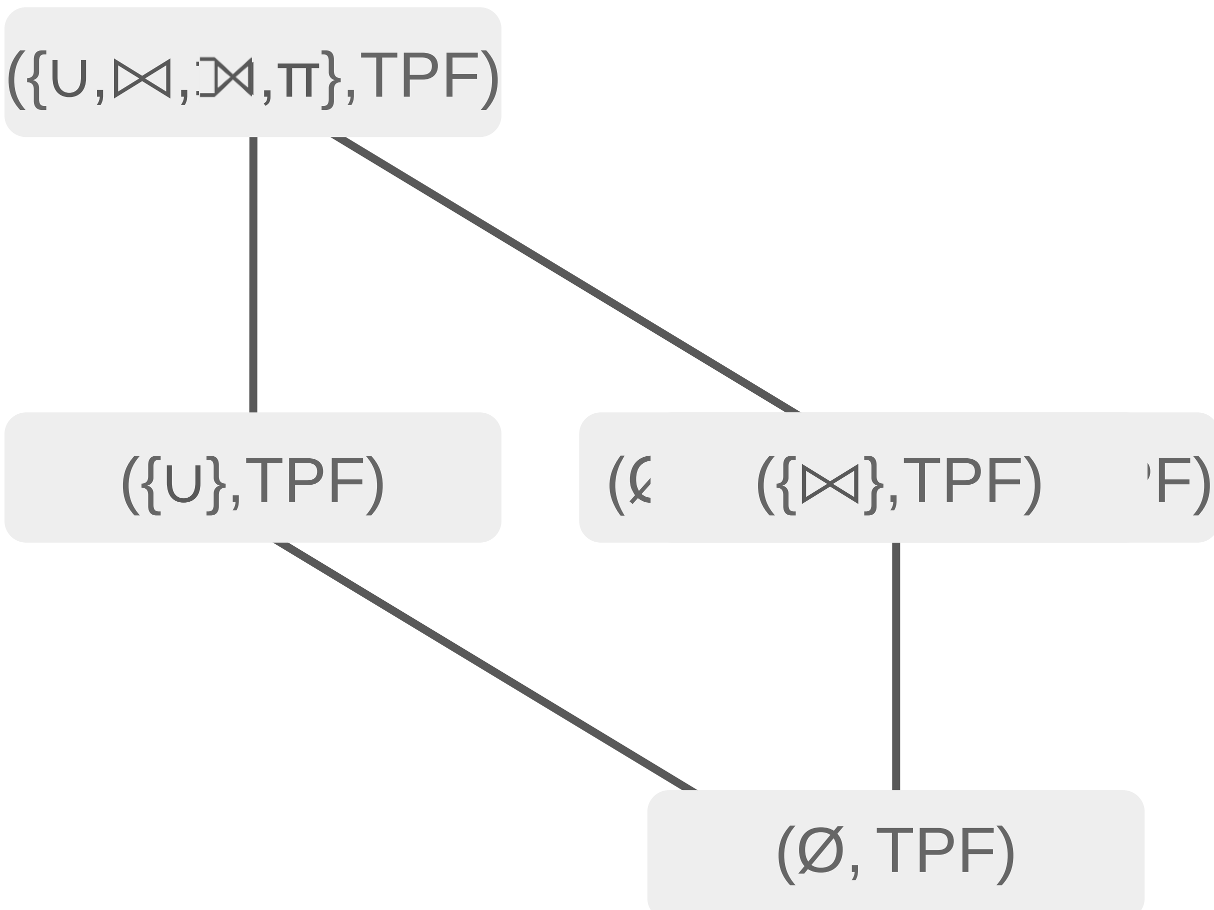
TPF

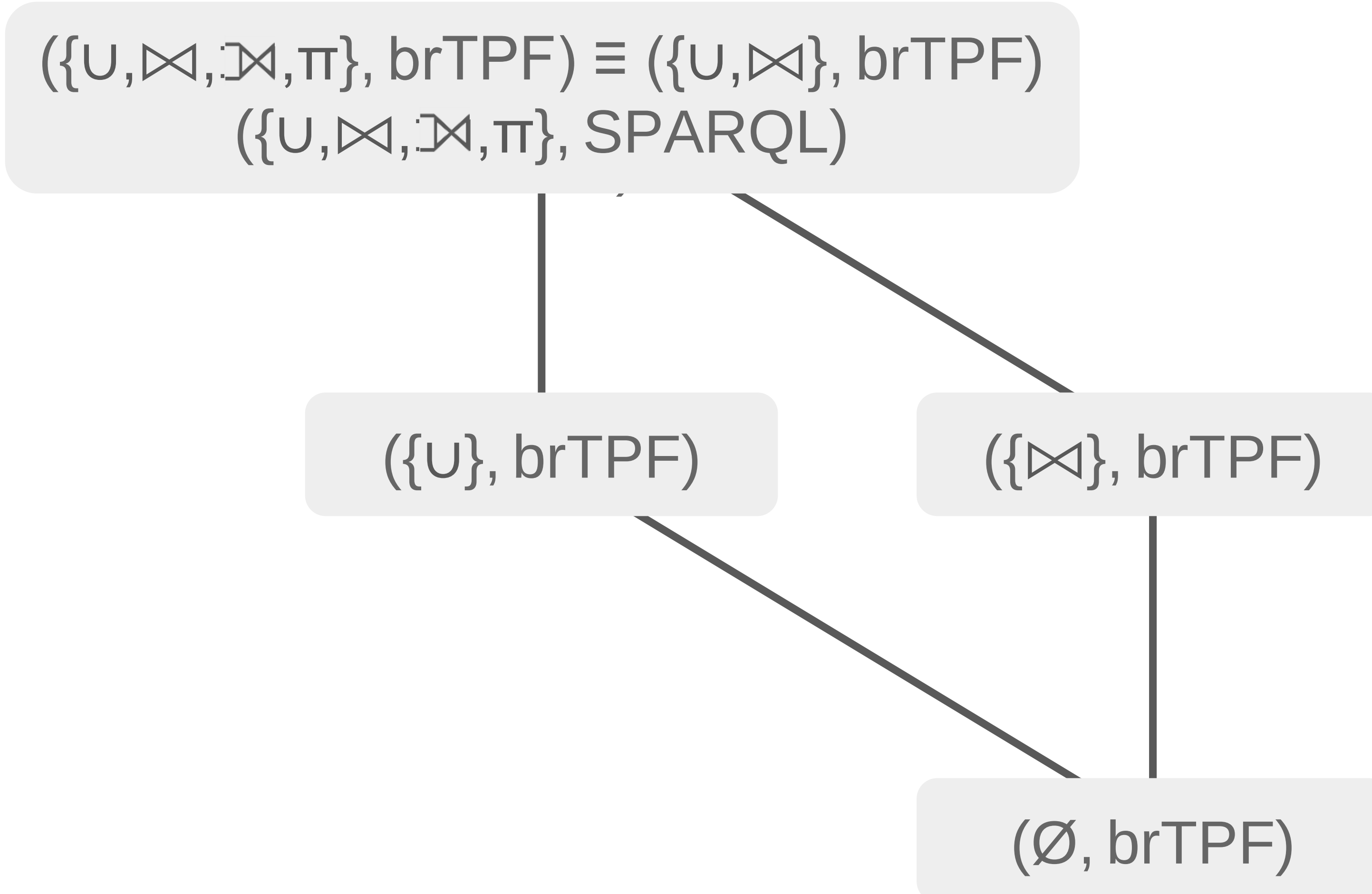
brTPF

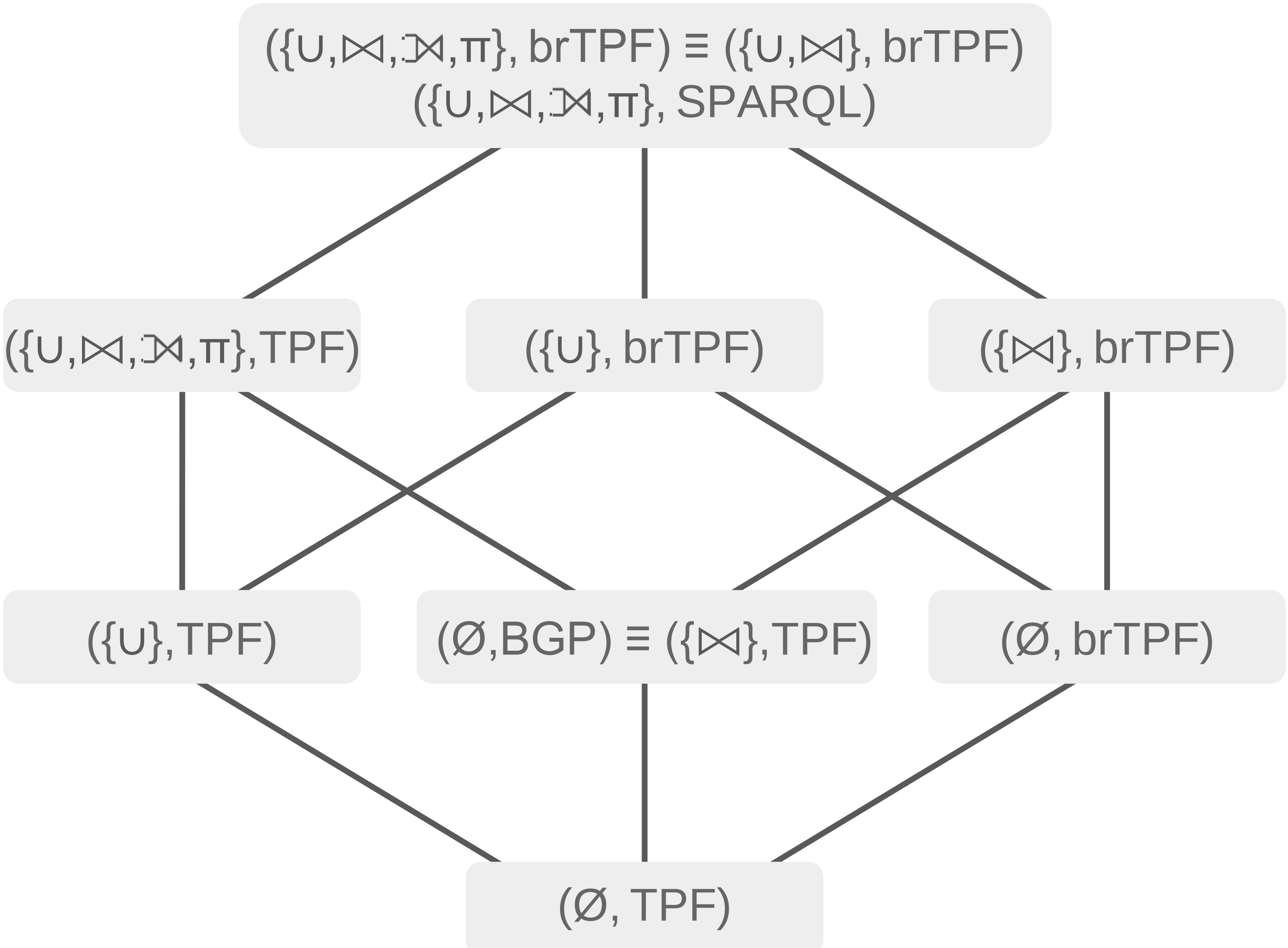
BGP

SPARQL

LDF interfaces

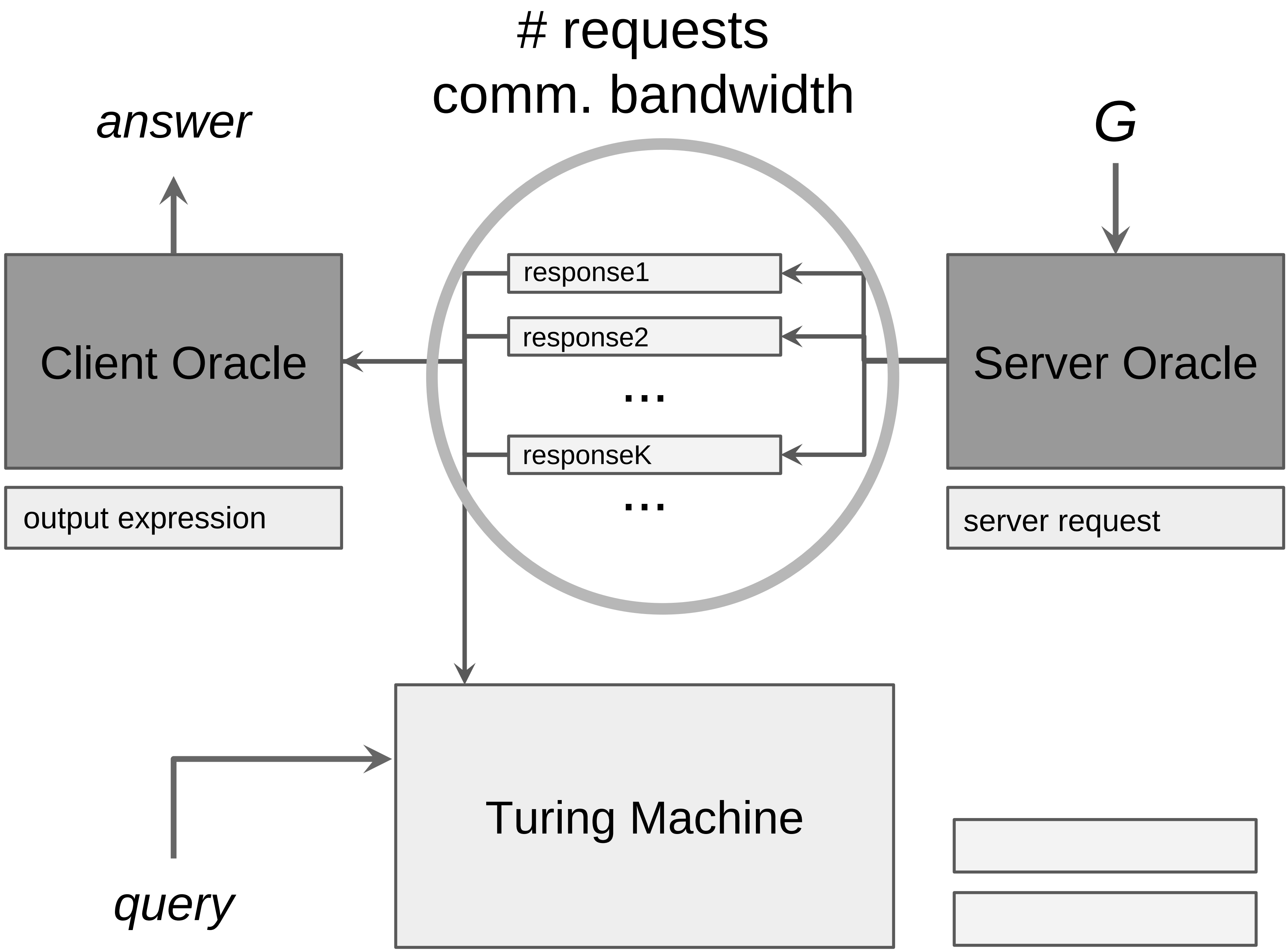

({∪,⋈,⟕,π},TPF)
({∪},TPF)
({⋈},TPF)
(Ø, TPF)


({∪,⋈,⟕,π}, brTPF) ≡ ({∪,⋈}, brTPF)
({∪,⋈,⟕,π}, SPARQL)
({∪}, brTPF)
({⋈}, brTPF)
(∅, brTPF)



({∪,⋈,⟕,π}, brTPF) ≡ ({∪,⋈}, brTPF)
({∪,⋈,⟕,π}, SPARQL)
({∪,⋈,⟕,π},TPF)
({∪}, brTPF)
({⋈}, brTPF)
({∪},TPF)
(Ø,BGP) ≡ ({⋈},TPF)
(Ø, brTPF)
(Ø, TPF)

Fine-Grained Complexity Analysis



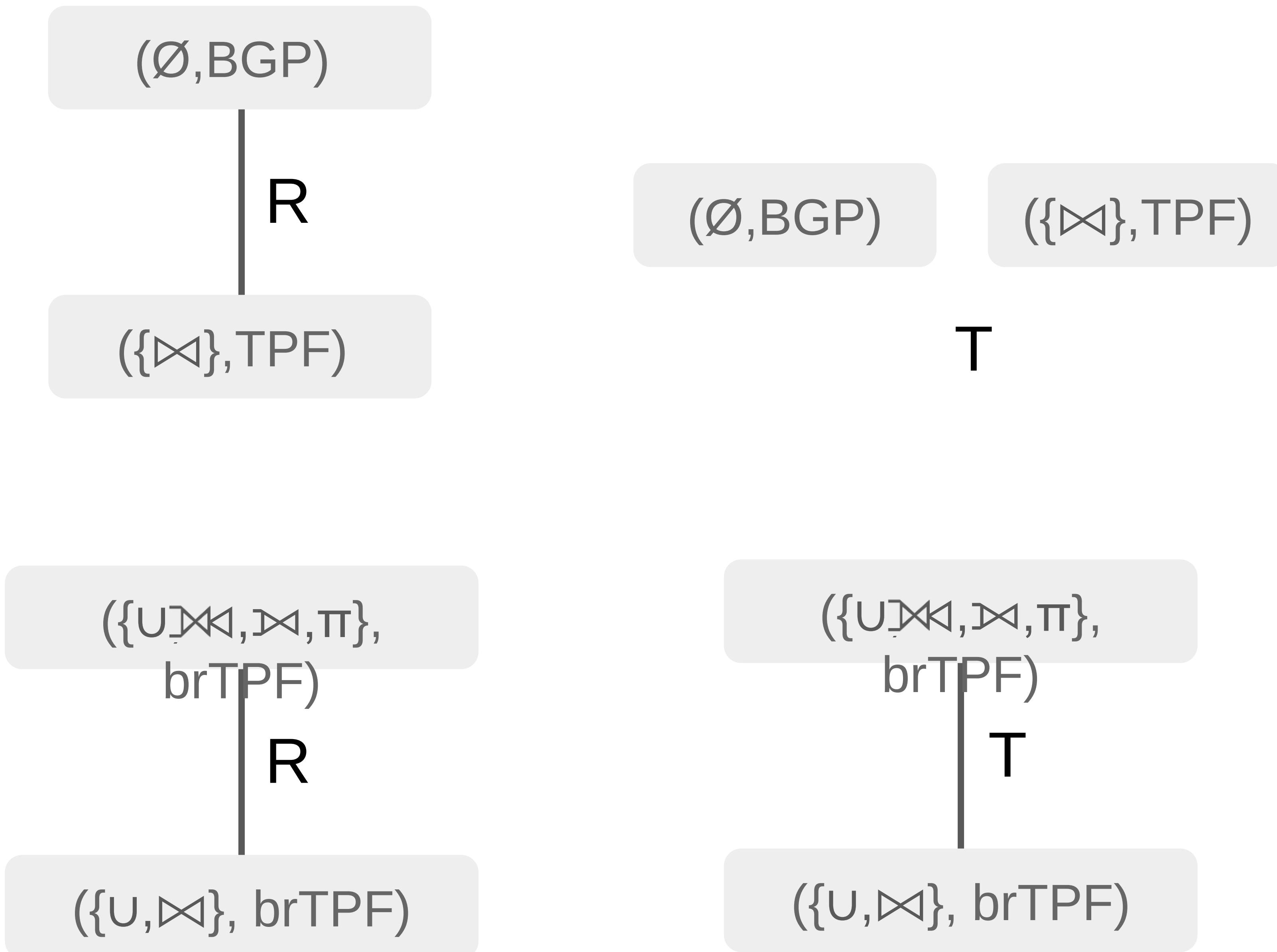
requests
comm. bandwidth
answer
G
Client Oracle
response1
response2
...
responseK
...
Server Oracle
output expression
server request
Turing Machine
query

$$(L_C L_S) \prec_{\mathrm{T}} (R_C R_S)$$

in terms of
|resp1| + |resp2| + … + |respK|

$$(L_C L_S) \prec_{\mathrm{R}} (R_C R_S)$$

in terms of K


(Ø,BGP)
R
({⋈},TPF)
(Ø,BGP)
({⋈},TPF)
T
({∪,⟕,⋈,π},
brTPF)
R
({∪,⋈}, brTPF)
({∪,⟕,⋈,π},
brTPF)
T
({∪,⋈}, brTPF)

A theory for comparing different access protocols for SemWeb data

More fundamental understanding of combinations of LDF interfaces

Machine model + first results on expressiveness and complexity

A theory for comparing different access protocols for SemWeb data

Include new LDF interfaces
and client languages

Improve the machine model
and consider new metrics

Formal study of SemWeb query planning

A Formal Framework to Compare Linked Data Fragments

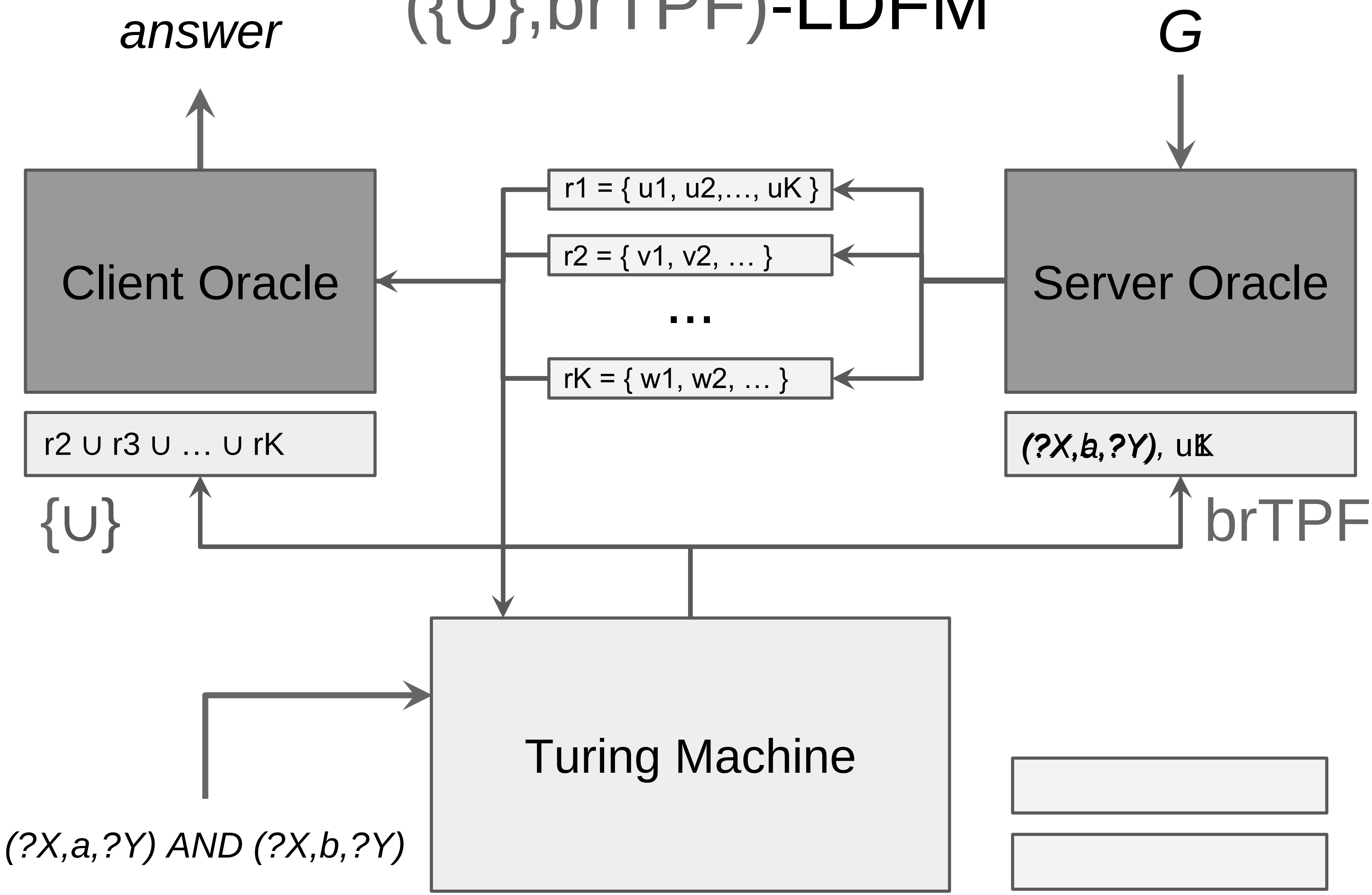
Olaf Hartig   Ian Letter   Jorge Pérez

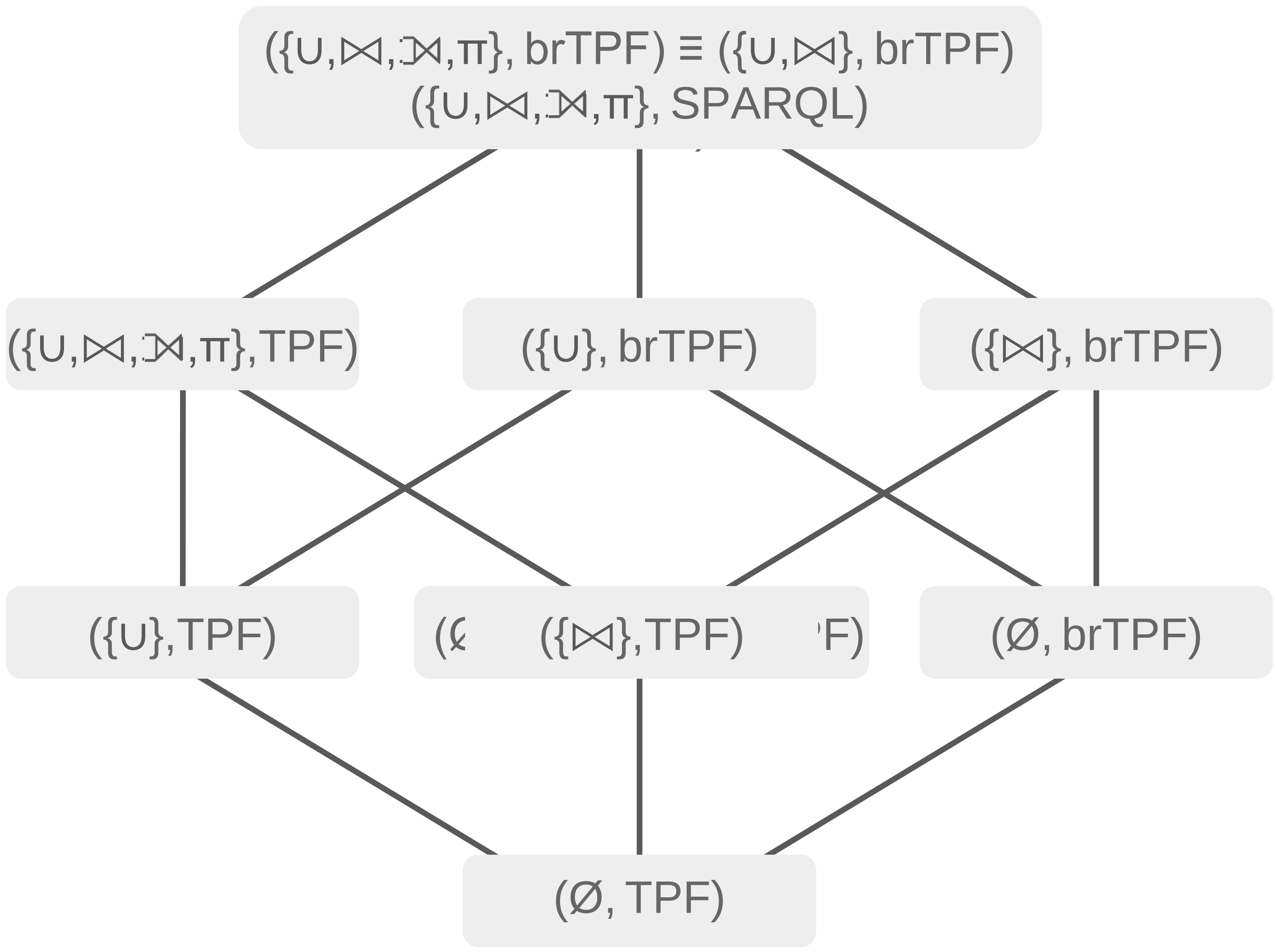
@olafhartig   @perez

Linköping University

Chilean Center for Semantic Web Research

({∪},brTPF)-LDFM


answer
G
Client Oracle
Server Oracle
r1 = { u1, u2,…, uK }
r2 = { v1, v2, … }
…
rK = { w1, w2, … }
r2 ∪ r3 ∪ … ∪ rK
(?X,a,?Y), u1
(?X,b,?Y), uK
{∪}
brTPF
Turing Machine
(?X,a,?Y) AND (?X,b,?Y)


({∪,⋈,⟕,π}, brTPF) ≡ ({∪,⋈}, brTPF)
({∪,⋈,⟕,π}, SPARQL)
({∪,⋈,⟕,π},TPF)
({∪}, brTPF)
({⋈}, brTPF)
({∪},TPF)
({⋈},TPF)
(Ø, brTPF)
(Ø, TPF)