Approximating equivalence queries through membership queries

Alexis PINSON
supervised by Adrien POMMELLET

Sommaire

Equivalence queries

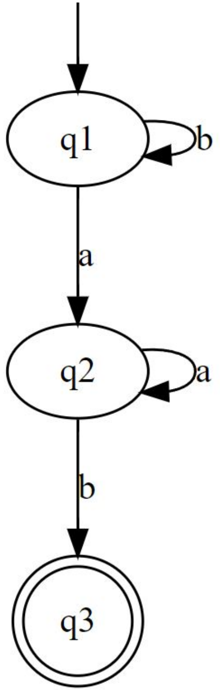
!=

Membership queries

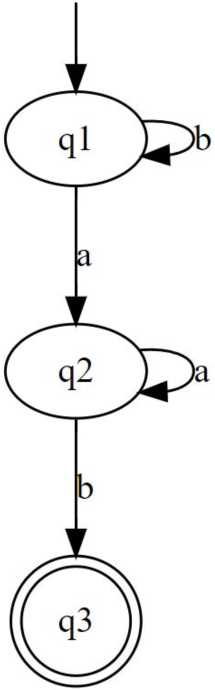
??

I - Mise en contexte

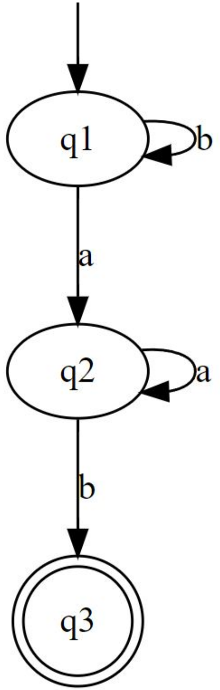
I - Mise en contexte

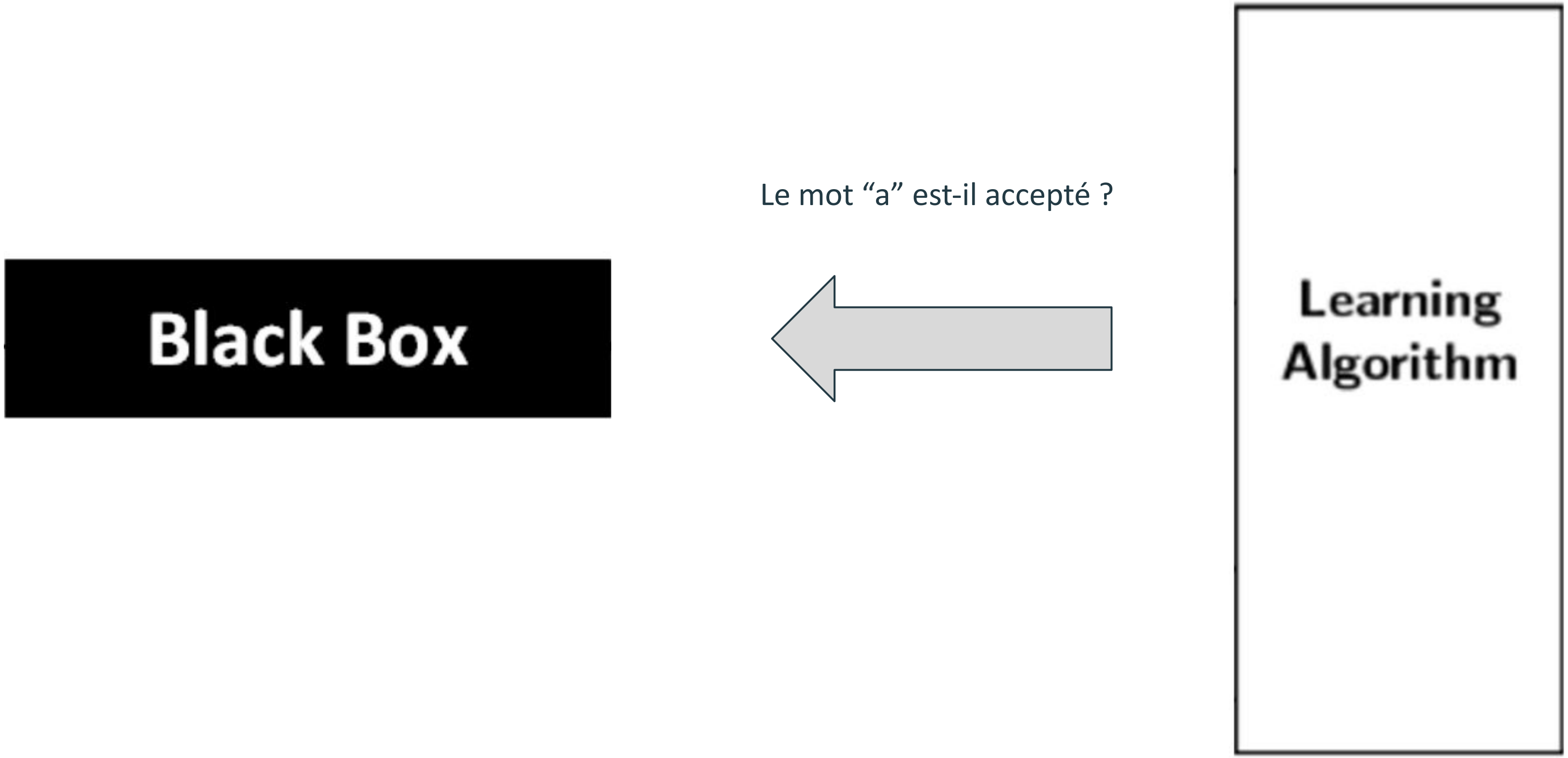


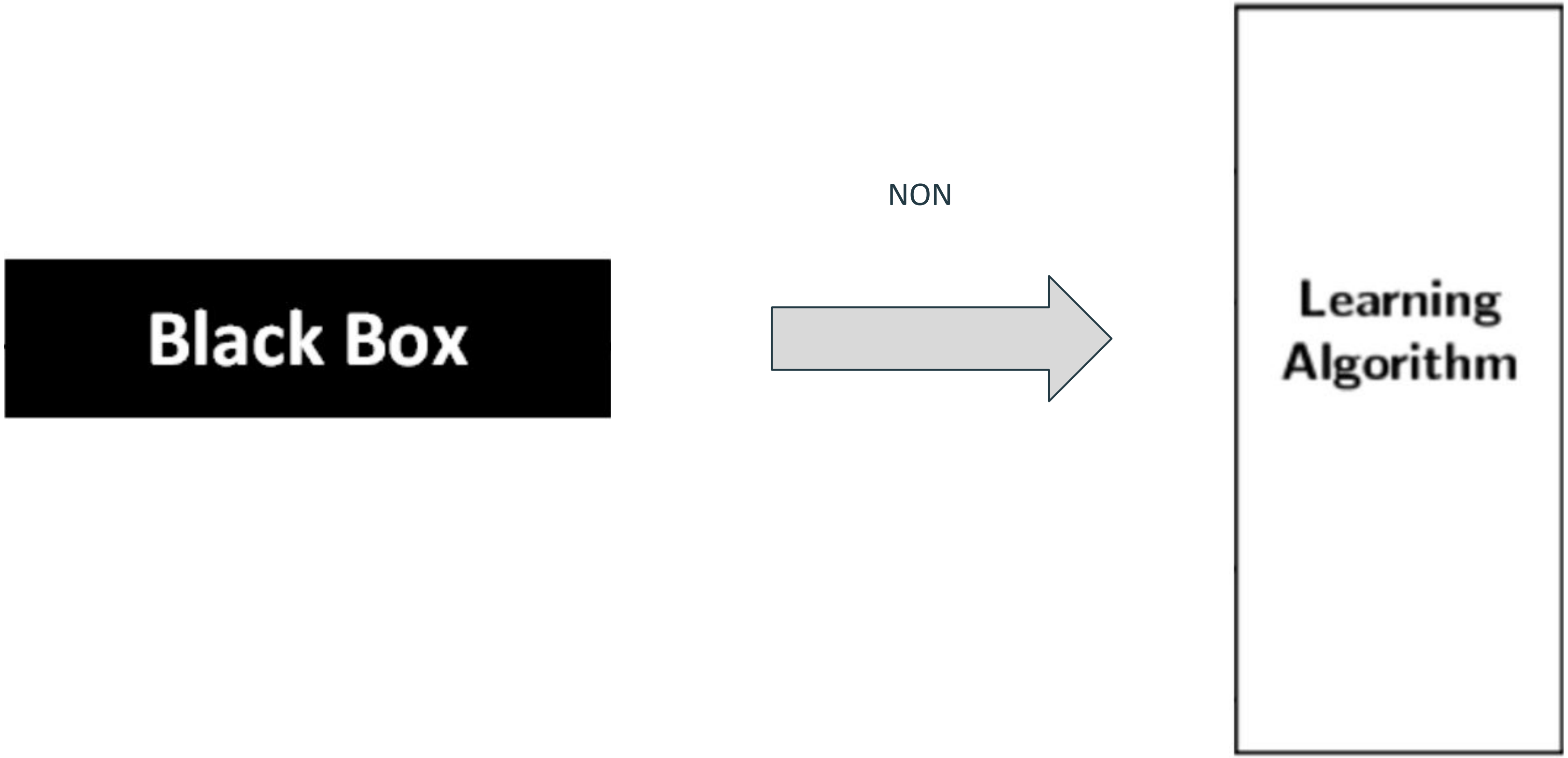
I - Mise en contexte

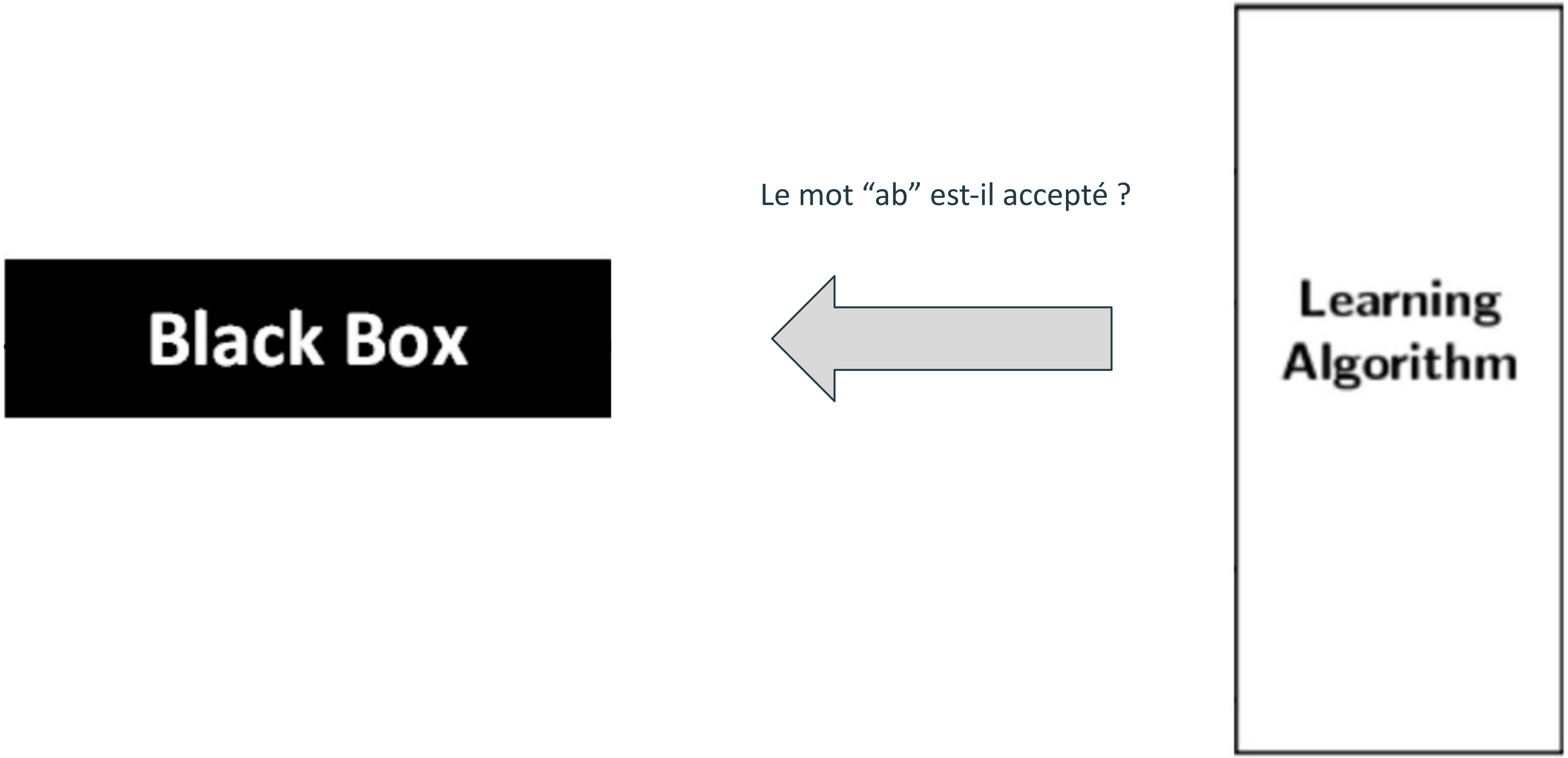


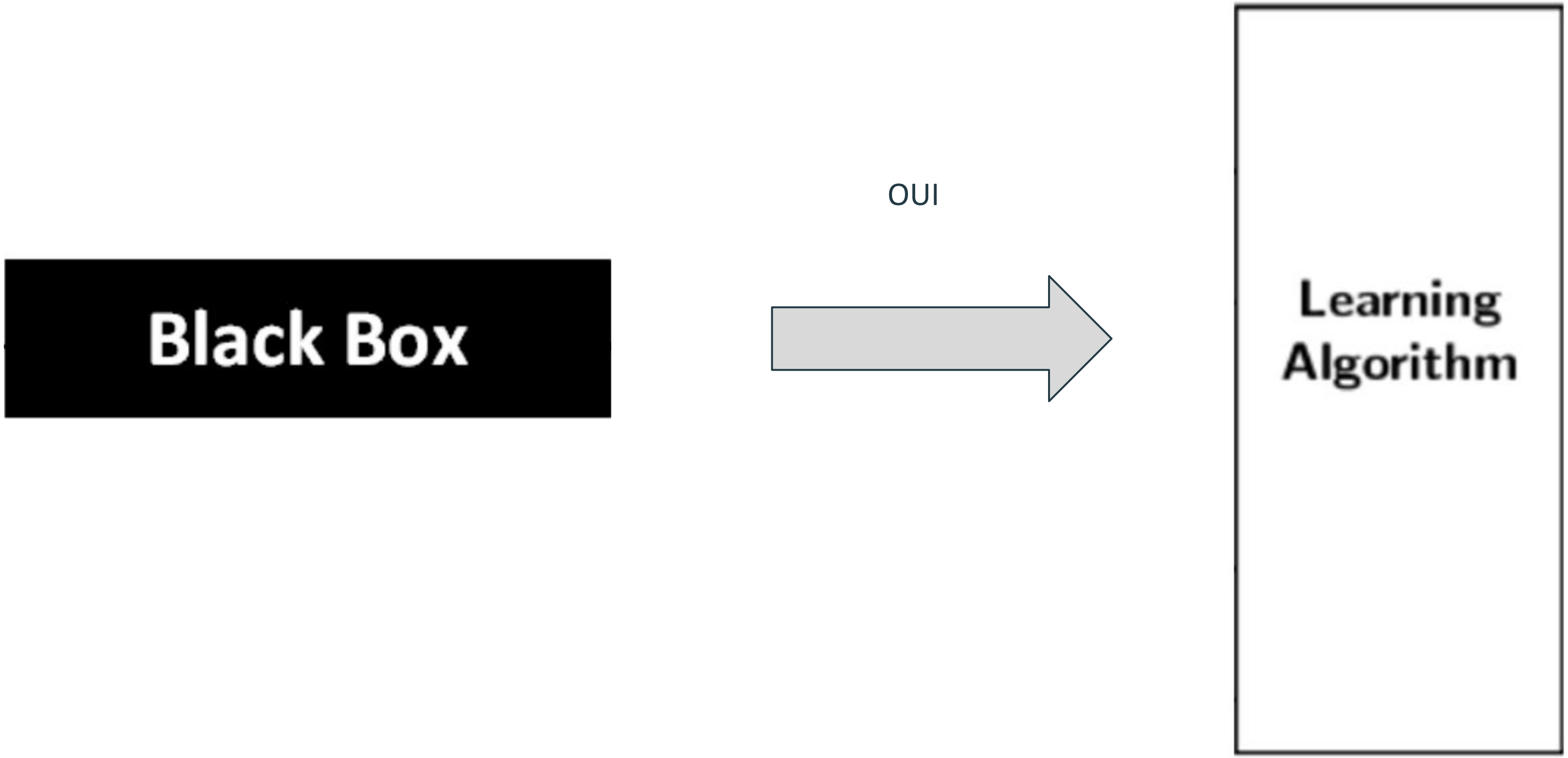
I - Mise en contexte

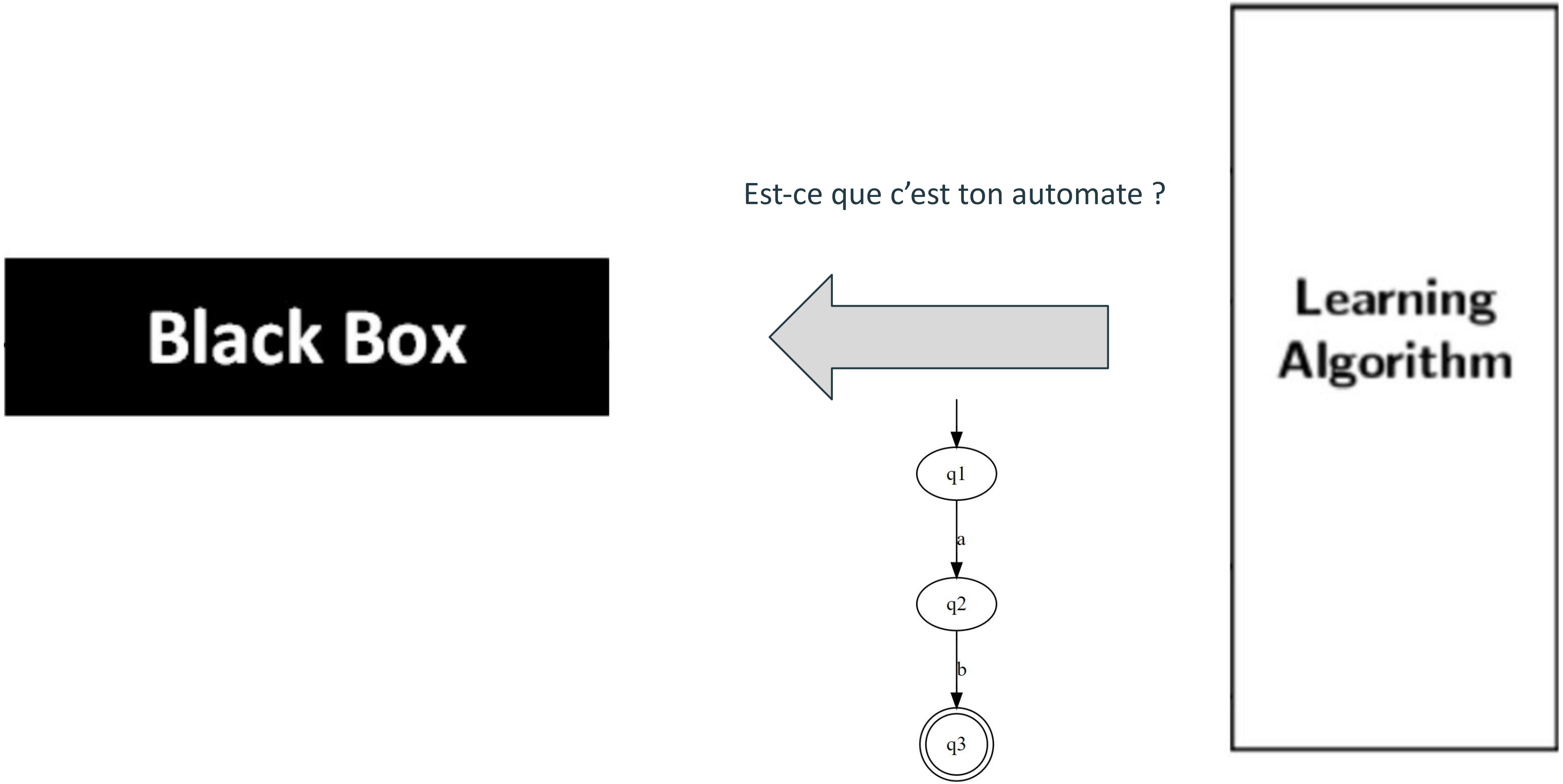


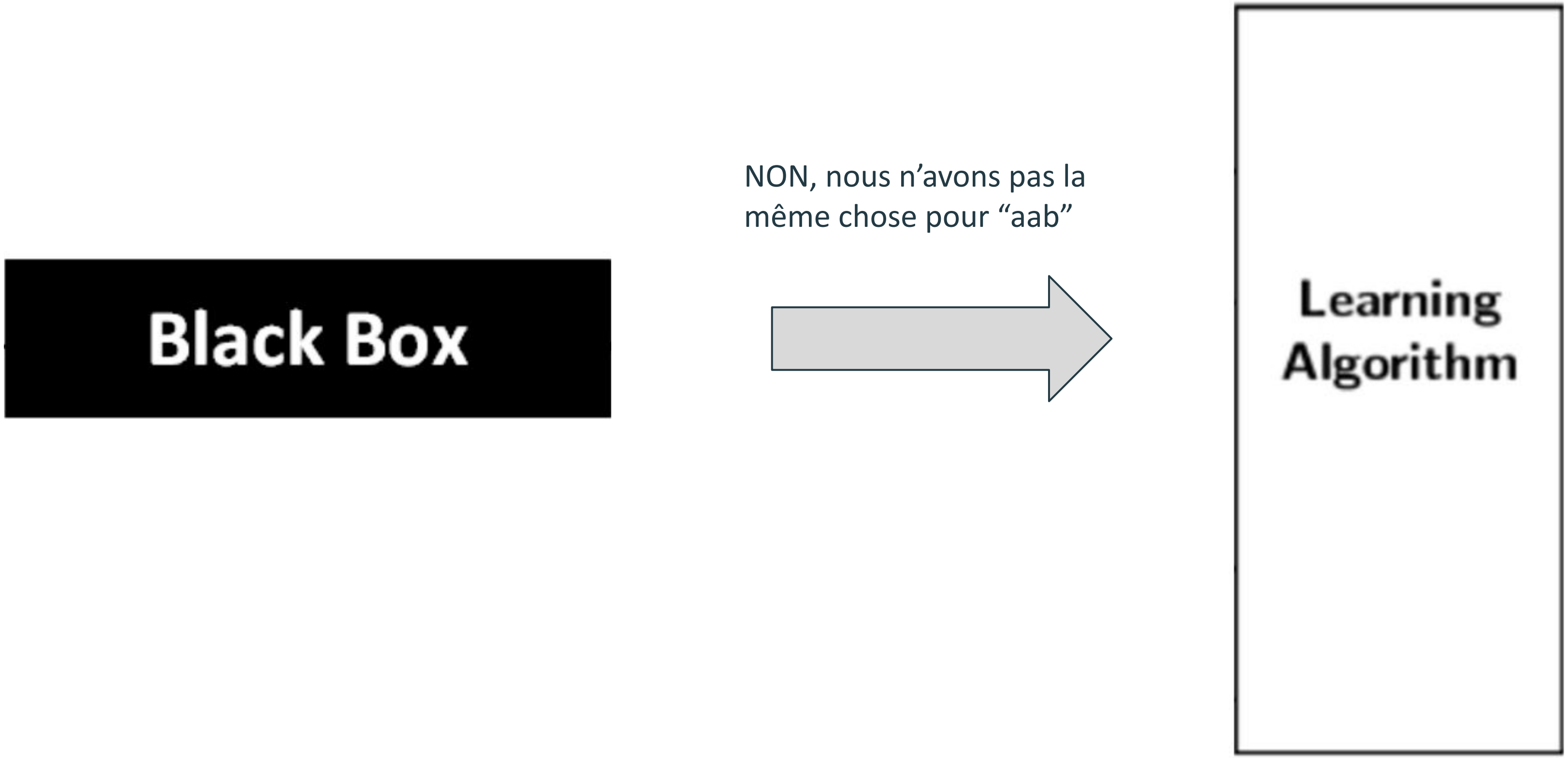

Le mot "a" est-il accepté ?
Black Box
Learning Algorithm


Black Box
NON
Learning Algorithm

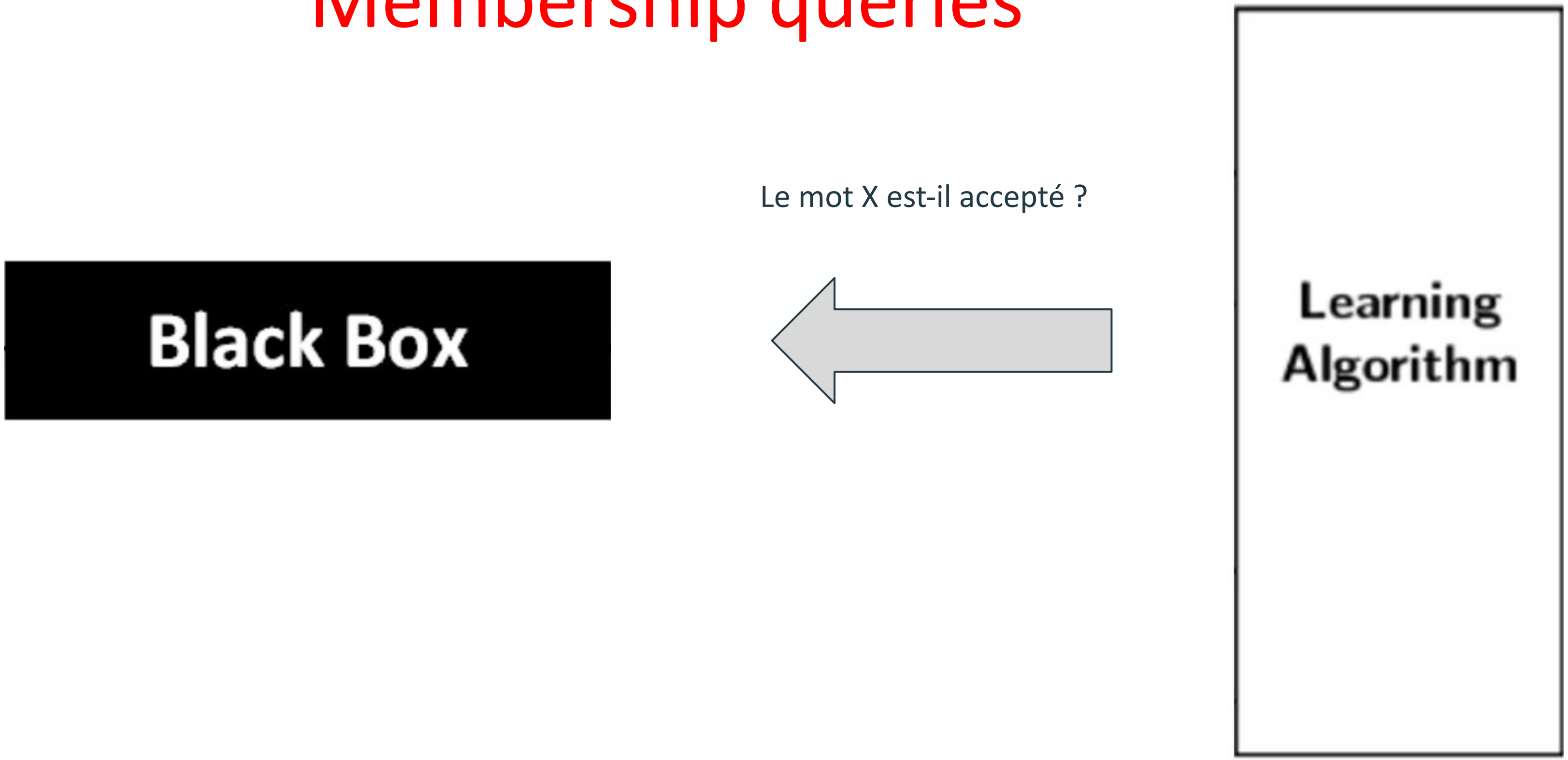

Le mot “ab” est-il accepté ?
Black Box
Learning
Algorithm


Black Box
OUI
Learning Algorithm

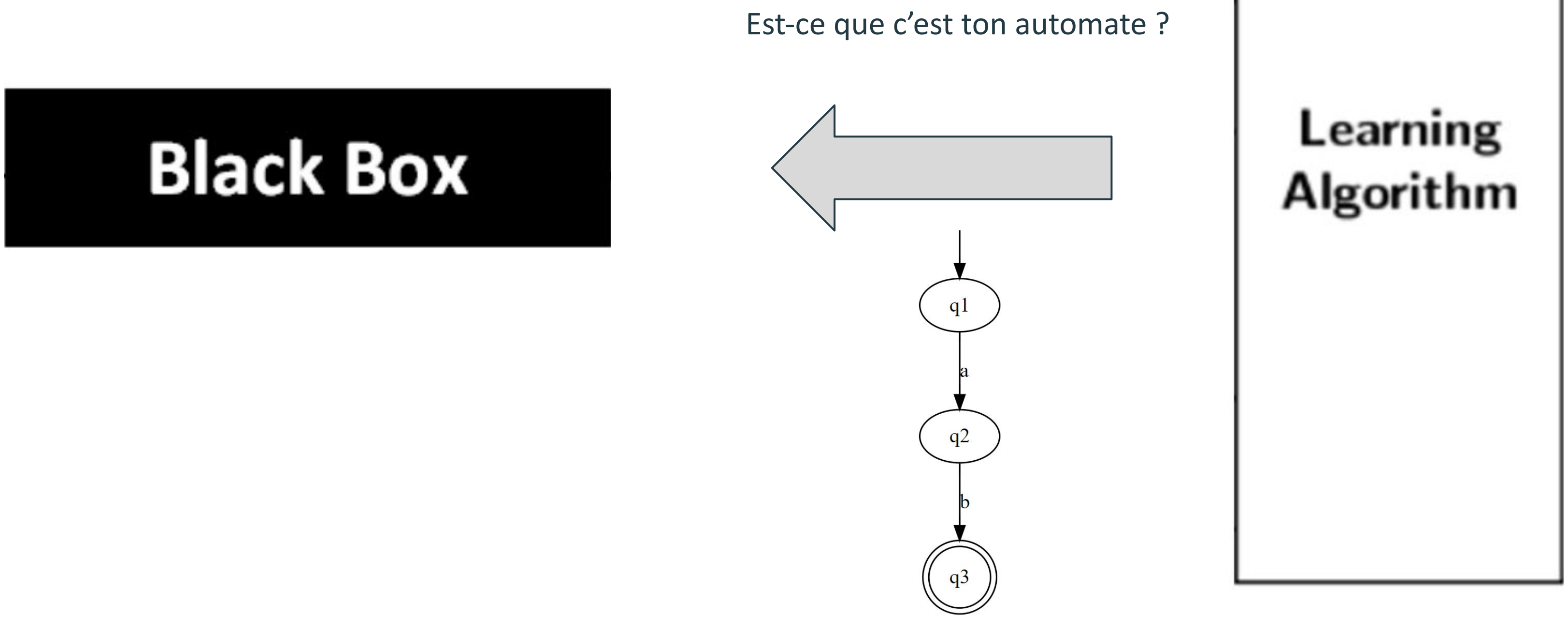

Black Box
Est-ce que c'est ton automate ?
q1
a
q2
b
q3
Learning
Algorithm

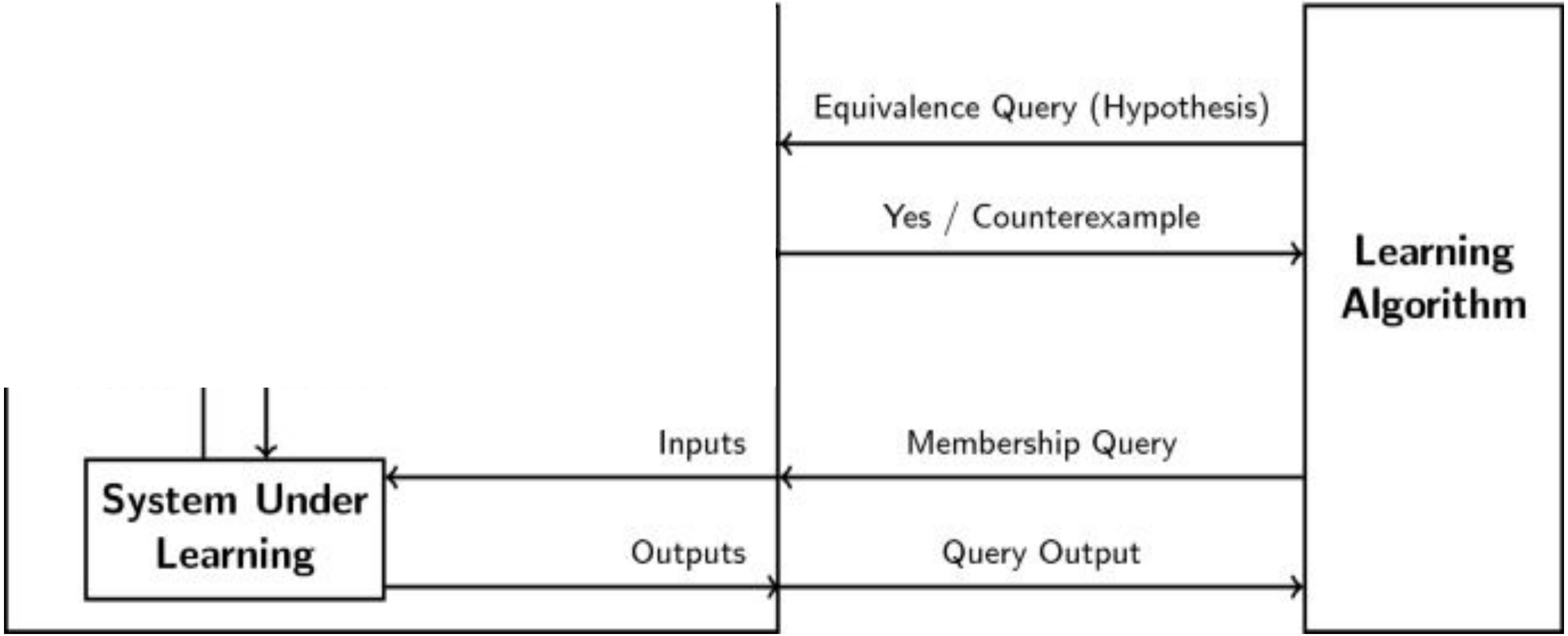

Black Box
NON, nous n’avons pas la
même chose pour “aab”
Learning
Algorithm

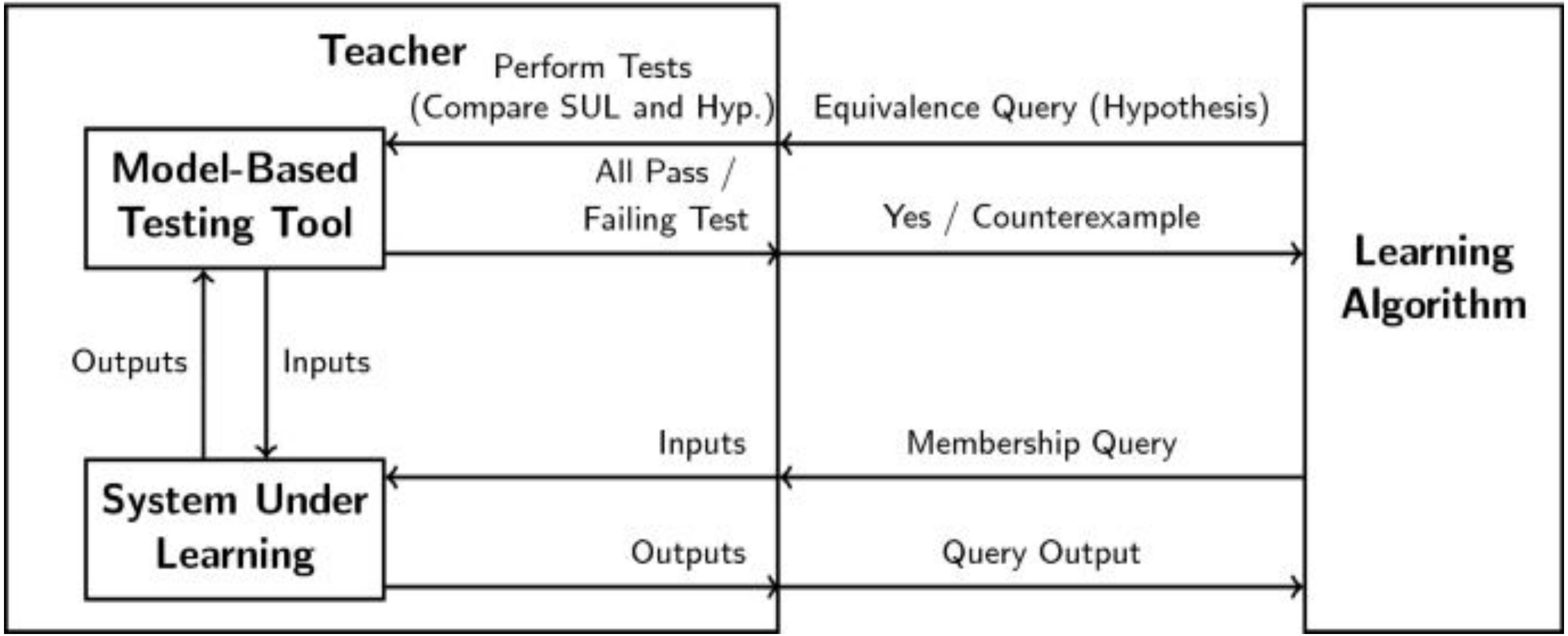
Membership queries



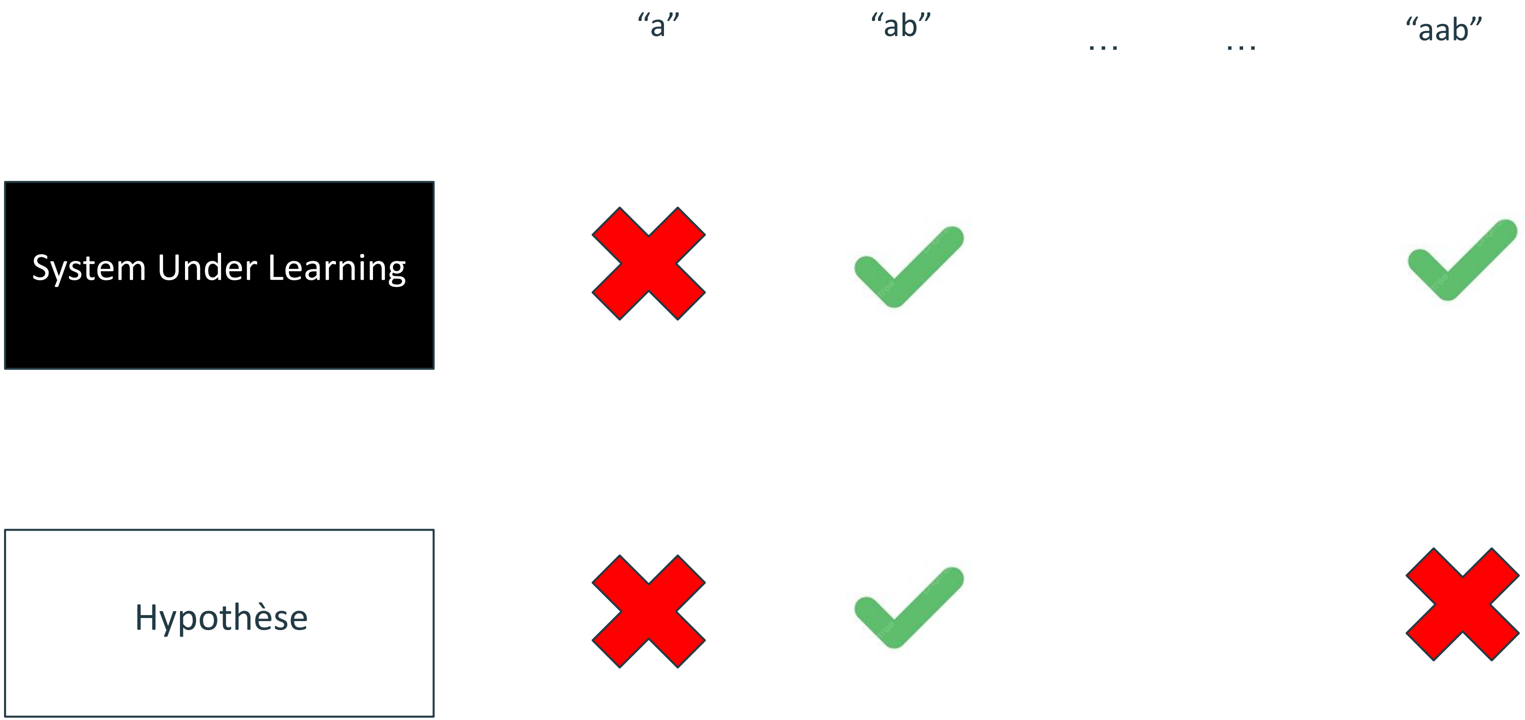
Equivalence queries



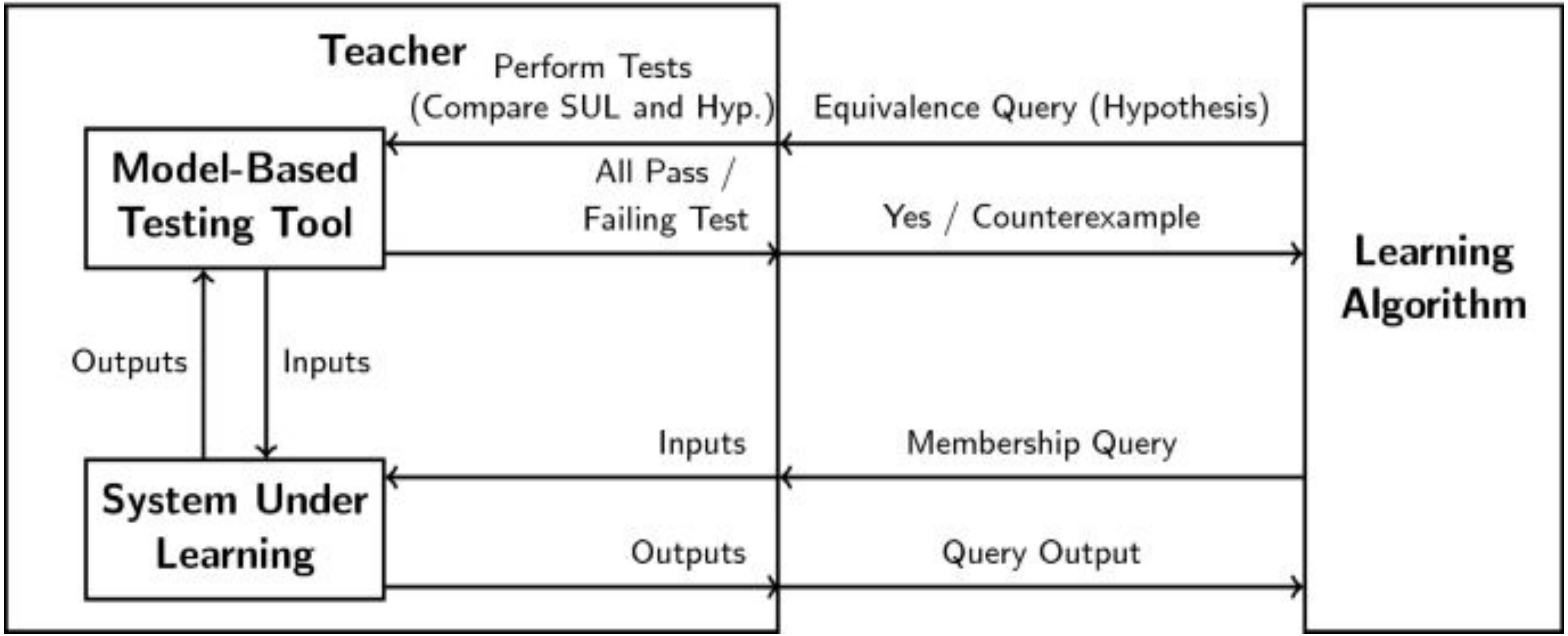

Equivalence Query (Hypothesis)
Yes / Counterexample
Learning Algorithm
Inputs
Membership Query
System Under Learning
Outputs
Query Output



Teacher
Perform Tests
(Compare SUL and Hyp.)
Equivalence Query (Hypothesis)
Model-Based Testing Tool
All Pass /
Failing Test
Yes / Counterexample
Learning Algorithm
Outputs
Inputs
Inputs
Membership Query
System Under Learning
Outputs
Query Output

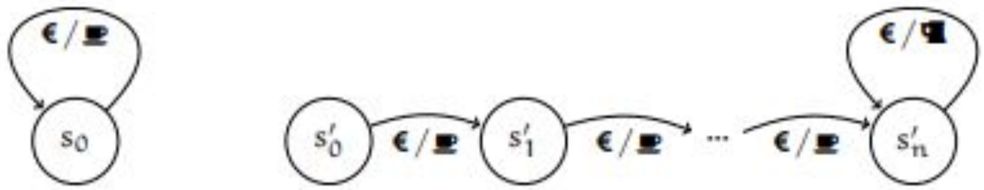


"a"
"ab"
…
…
"aab"
System Under Learning
Hypothèse

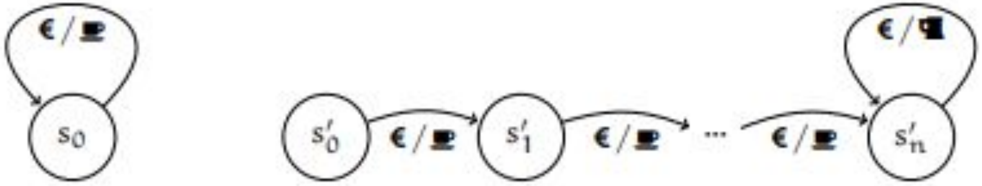

Teacher
Perform Tests
(Compare SUL and Hyp.)
Equivalence Query (Hypothesis)
Model-Based Testing Tool
All Pass /
Failing Test
Yes / Counterexample
Learning Algorithm
Outputs
Inputs
Inputs
Membership Query
System Under Learning
Outputs
Query Output

Est-ce que c’est possible ?

Est-ce que c'est possible ?



Est-ce que c’est possible ?



“Approximating equivalence queries through membership queries”

II - Génération de tests

II - Génération de tests

W-method (Chow, 1978 and Vasilevskii, 1973)

II - Génération de tests

W-method (Chow, 1978 and Vasilevskii, 1973)

A set of sequences W is a called a *characterisation set* if it contains a separating sequence for each pair of inequivalent states in M.

II - Génération de tests

W-method (Chow, 1978 and Vasilevskii, 1973)

A set of sequences W is a called a *characterisation set* if it contains a separating sequence for each pair of inequivalent states in M.

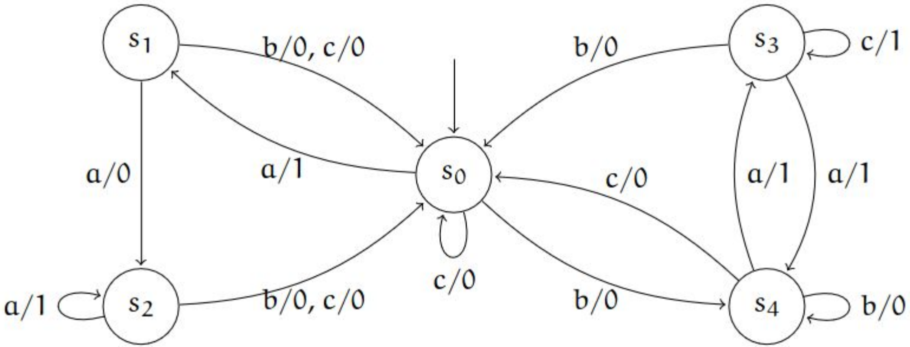
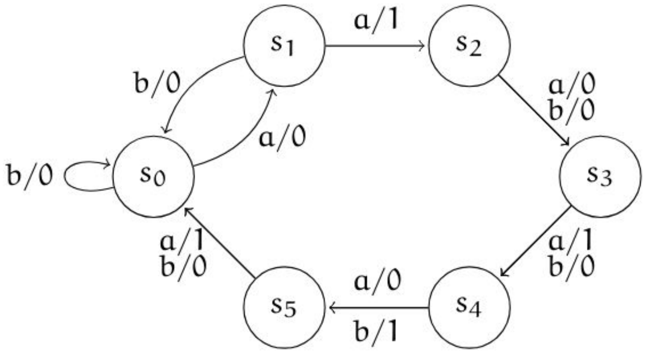


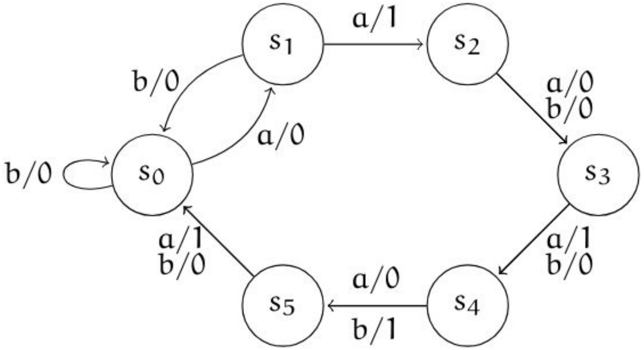
Figure 2.1 An example specification with input $I = \{a, b, c\}$ and output $O = \{0, 1\}$.

Approche minimale

Approche minimale



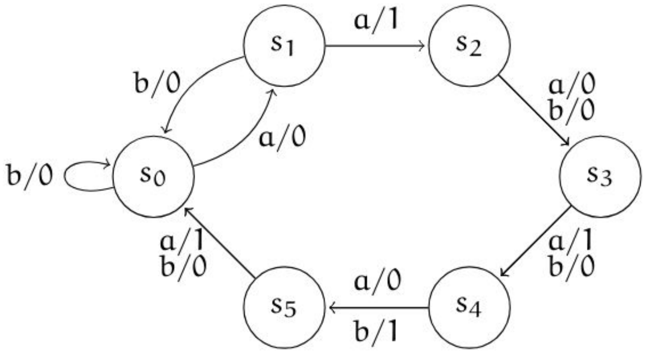
Approche minimale

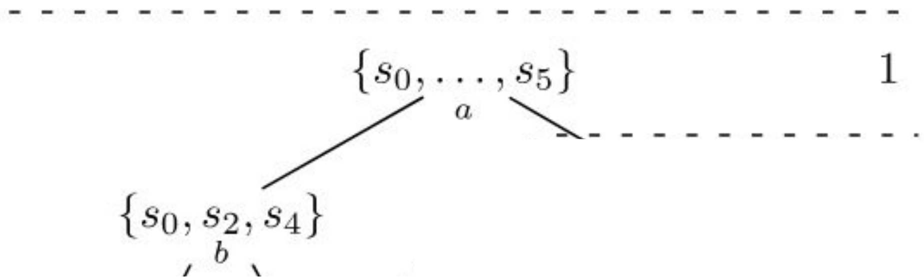


$\{s_0, \ldots, s_5\}$ 1

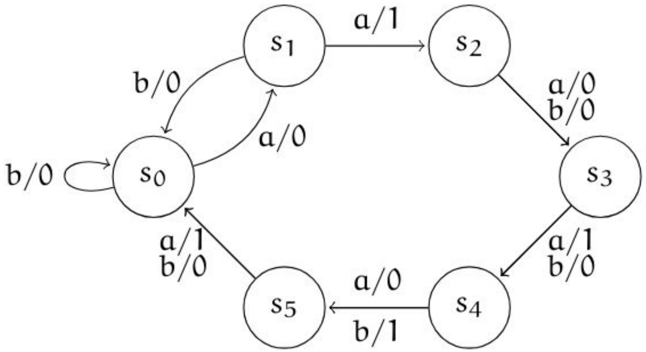
a

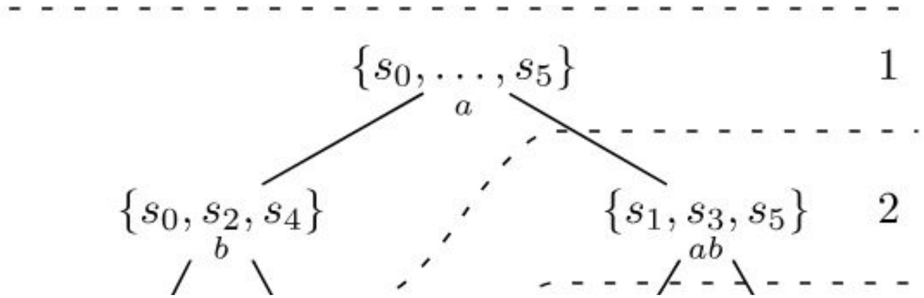
Approche minimale



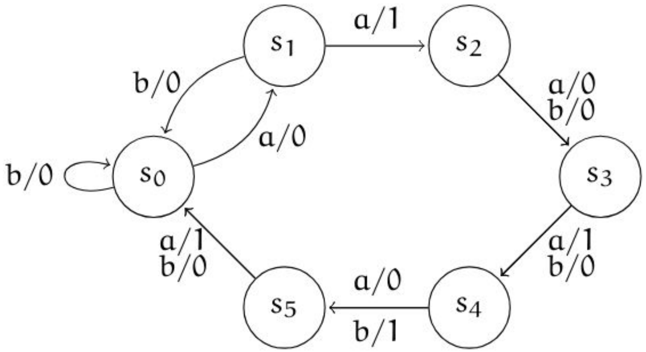


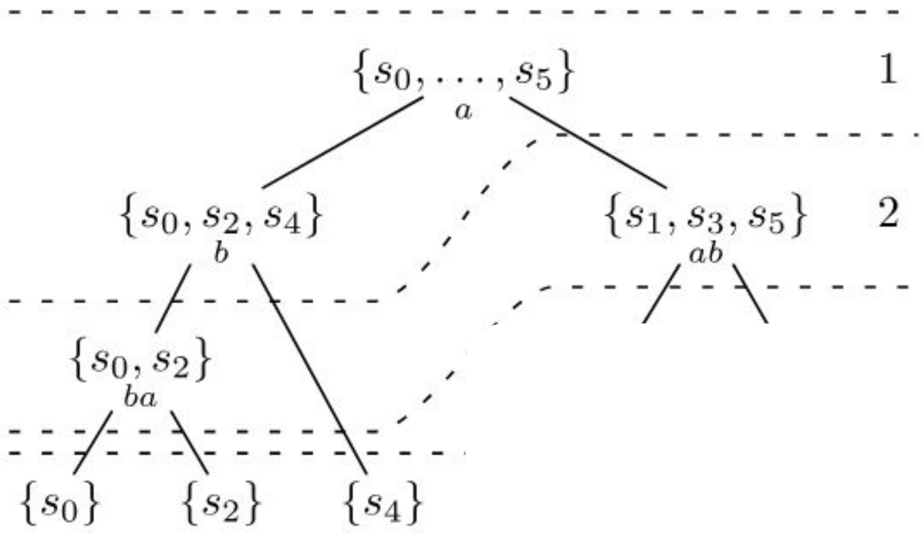
Approche minimale



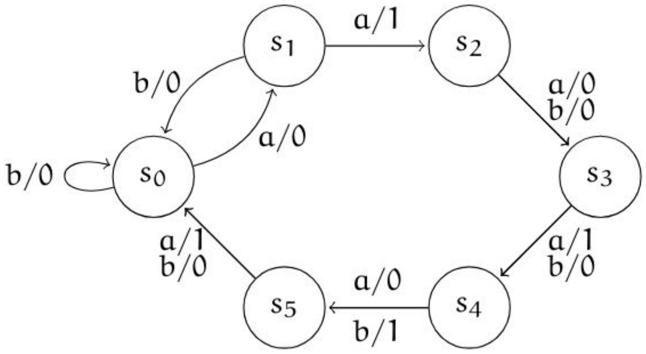


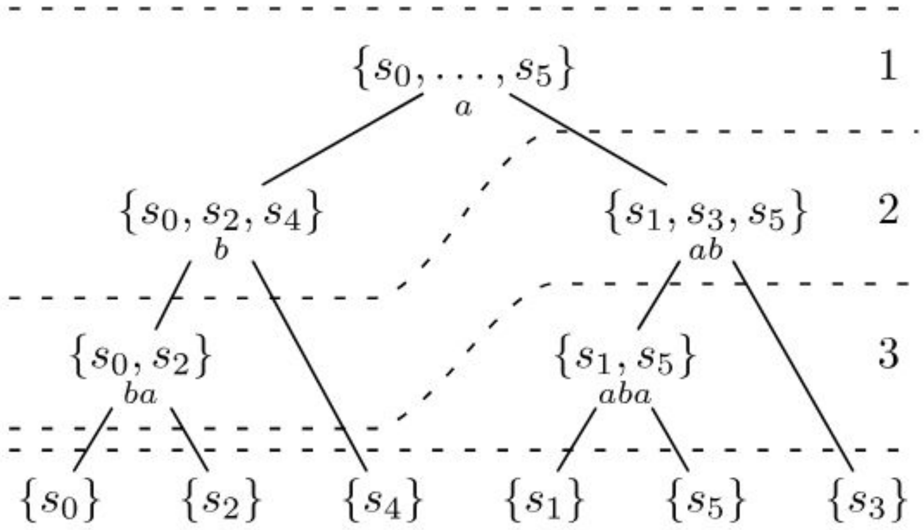
Approche minimale





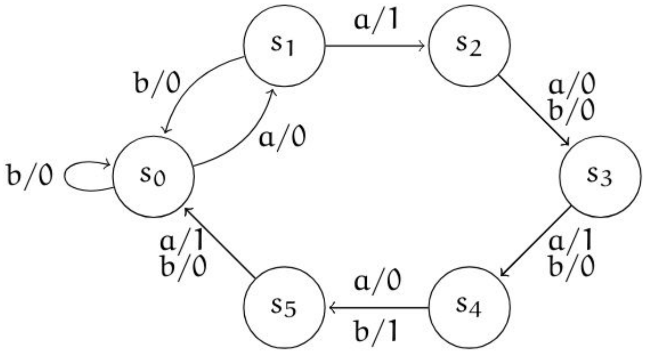
Approche minimale

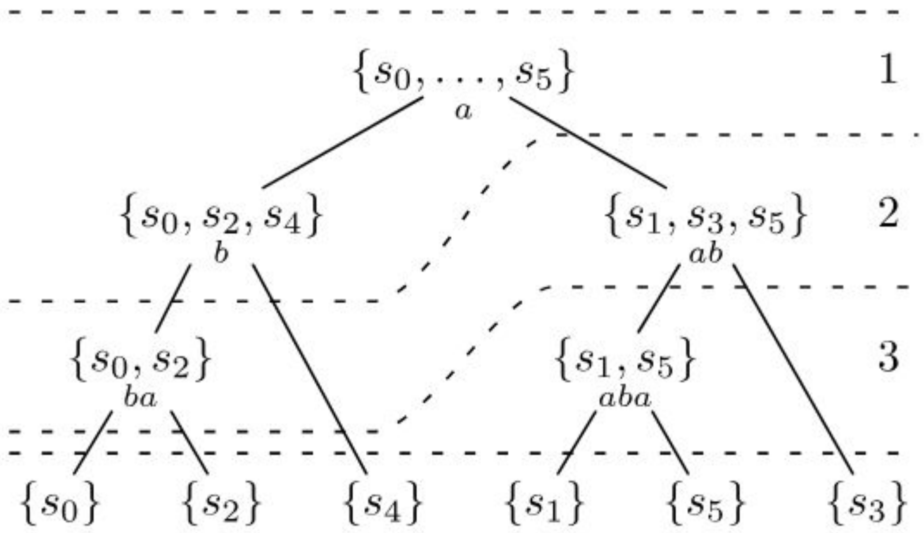




Approche minimale

suite de tests minime





Approche learner aware

Approche learner aware

Learning algorithm → Fournit W → Trouve le plus petit sous-ensemble de W qui reste un characterisation set → Produire les tests à partir de ce nouveau W

Approche learner aware

Learning algorithm

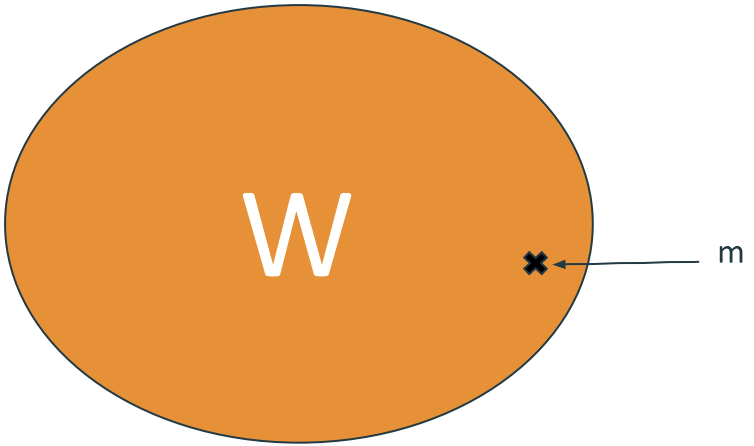
Fournit W

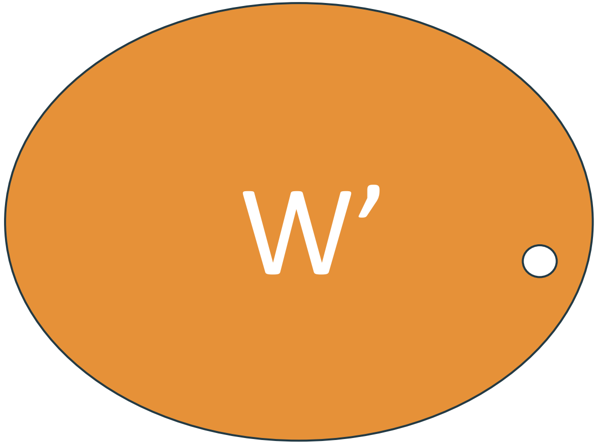
Trouve le plus petit sous-ensemble de W qui reste un characterisation set

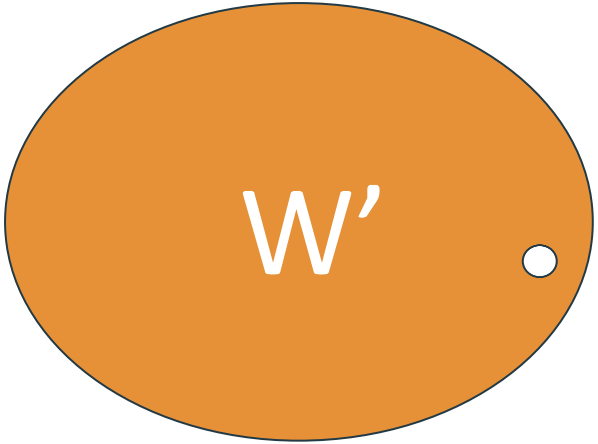
Produire les tests à partir de ce nouveau W

tests qui utilisent un maximum le cache donc peu coûteux si équivalence

W

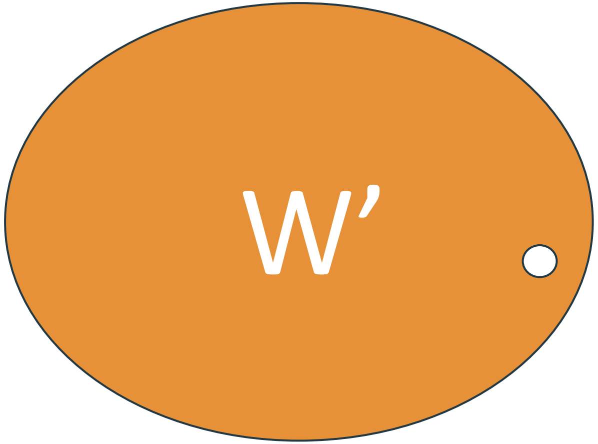

W
m


W’



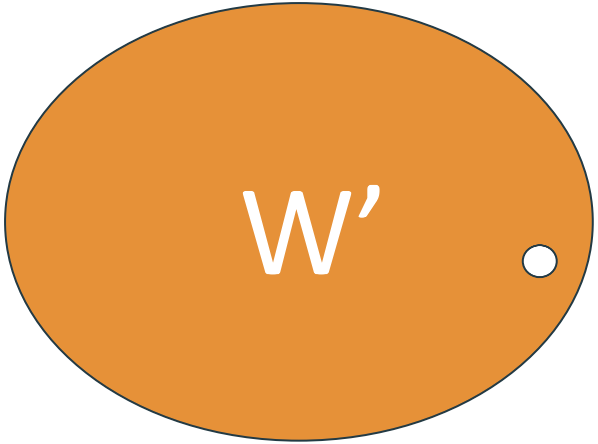
Est-ce qu’il reste un characterisation set ?

Oui : le mot m n’est donc pas obligatoire



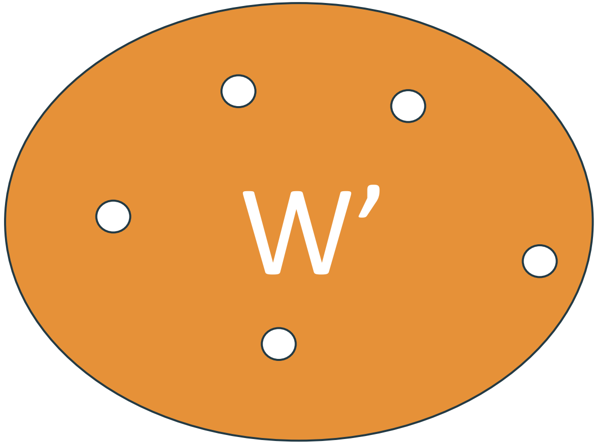
Est-ce qu’il reste un characterisation set ?

Oui : le mot m n’est donc pas obligatoire



Non : le mot m est donc obligatoire

Est-ce qu’il reste un characterisation set ?


W’

III - Résultats et conclusion

	Size 5 DFA (k=3)	Size 10 DFA (k=5)
Approche minimale (totale)	231	465
Approche learner aware (totale)	148	469
Approche minimale (équivalence)	216	437
Approche learner aware (équivalence)	138	446

Small Test Suites for Active Automata Learning*

Loes Kruger, Sebastian Junges, and Jurriaan Rot

Institute for Computing and Information Sciences,
Radboud University, Nijmegen, the Netherlands
{loes.kruger,sebastian.junges,jurriaan.rot}@ru.nl

5 Test Case Prioritization

To establish equivalence, *all* tests in a complete test suite need to be executed and their order is then irrelevant. However, to find a counterexample, we only need to execute tests until we hit that counterexample. This means that different orderings lead to significant performance changes [7]. In this section, we first describe the state-of-the-art in (ordered) test suites. We then create new, ordered test suites, that combine the ETS's from Sec. 4 adaptively.

Sources:

- Nominal Techniques and Black Box Testing for Automata Learning, Joshua Moerman
- Hopcroft, J. E. (1971). An n log n algorithm for minimizing states in a finite automaton. In Theory of Machines and Computations
- Moore, E. F. (1956). Gedanken–experiments on Sequential Machines
- Chow, T. S. (1978). Testing Software Design Modeled by Finite-State Machines
- Vasilevskii, M. P. (1973). Failure diagnosis of automata. Cybernetics and Systems Analysis
- Lee, D. & Yannakakis, M. (1994). Testing Finite-State Machines: State Identification and Verification
- Small Test Suites for Active Automata Learning (2024). Loes Kruger, Sebastian Junges, Jurriaan Rot

Algorithm 1 Constructing a W set learner aware

Require: A W set given by the learning algorithm

Ensure: A minimal subset of W which is still a characterisation set

```
function WFROM(W, K = ∅)
    Create a set C for all sets that are still characterisation set
    Create a set N for all states that are mandatory
    for all words w in W do
        construct F = W\{w}
        if F is a characterisation set then
            add F to C
        else
            add w to N
        end if
    end for
    if C = ∅ then
        return K ∪ N
    end if
    for all sets Wi in C do
        invoke WFROM(Wi, K ∪ N)
    end for
    return the WFROM with the smallest length
end function
```