cetic
Your Connection to ICT Research

ScalaTest & ScalaCheck au service d'OscaR.CBLS

Arthur ATTOUT

Stagiaire au département Optimisation

TIM de fin de stage, Charleroi, 19/04/2019

Stage encadré par Renaud De Landtsheer

www.enmieux.be

FEDER
UNION EUROPEENNE
Wallonie
LE FONDS EUROPÉEN DE DÉVELOPPEMENT RÉGIONAL
ET LA WALLONIE INVESTISSENT DANS VOTRE AVENIR

official partner | digital wallonia .be

Centre d'Excellence en **Technologies** de l'**Information** et de la **Communication**
www.cetic.be

Contexte

- Hénallux
- Master en Architecture des Systèmes Informatiques

- **Objectifs du stage**
  - Prise en main d'OscaR
  - Rédaction de tests unitaires

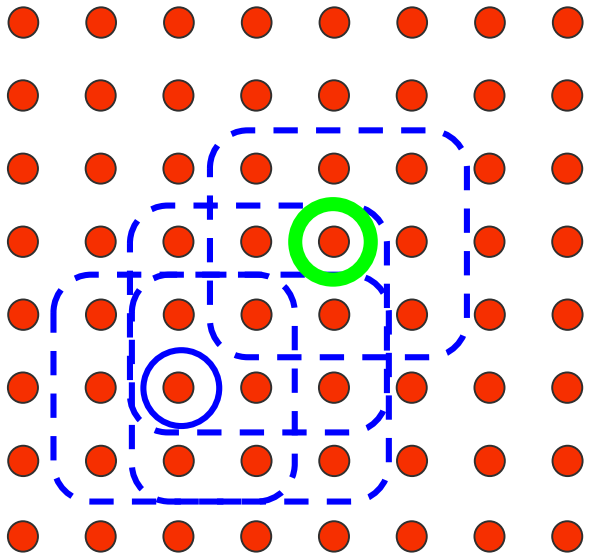
Préambule : OscaR

- OscaR : Framework de recherche opérationnelle

  - Problèmes d'optimisation
    - Routage, scheduling, bin-packing, …
  - Scala
  - Open-source

Préambule : CBLS

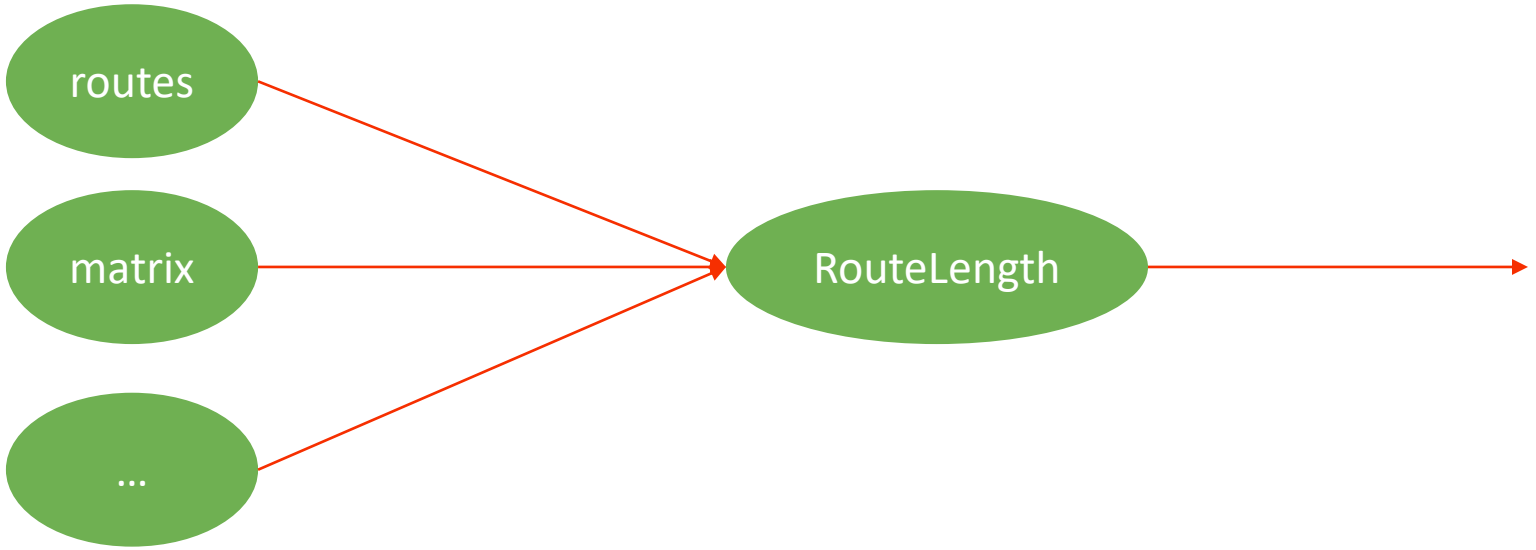
- **C**onstraint-**B**ased **L**ocal **S**earch
  - Partir d'une solution « au hasard »
  - Exploration du voisinage
  - Évaluer le voisinage => **fonction objective**



Point dans l'espace de recherche

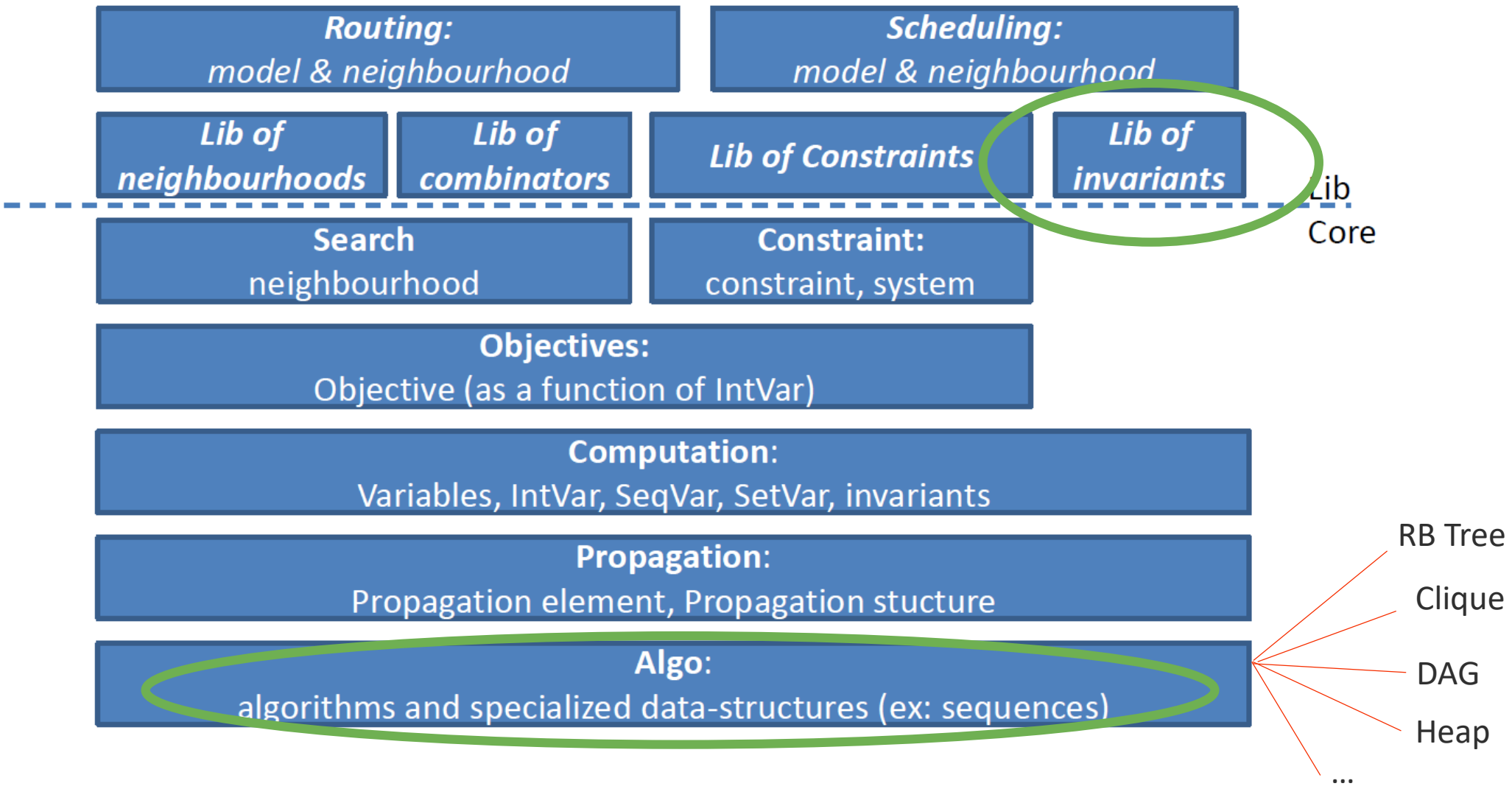
Préambule : Invariants

- La fonction objective est exprimée par des **invariants**
- **Invariant =** Maintien une/des valeur(s) de sortie par rapport à une/des valeur(s) d'entrée
- Exemple = `RouteLength( )`



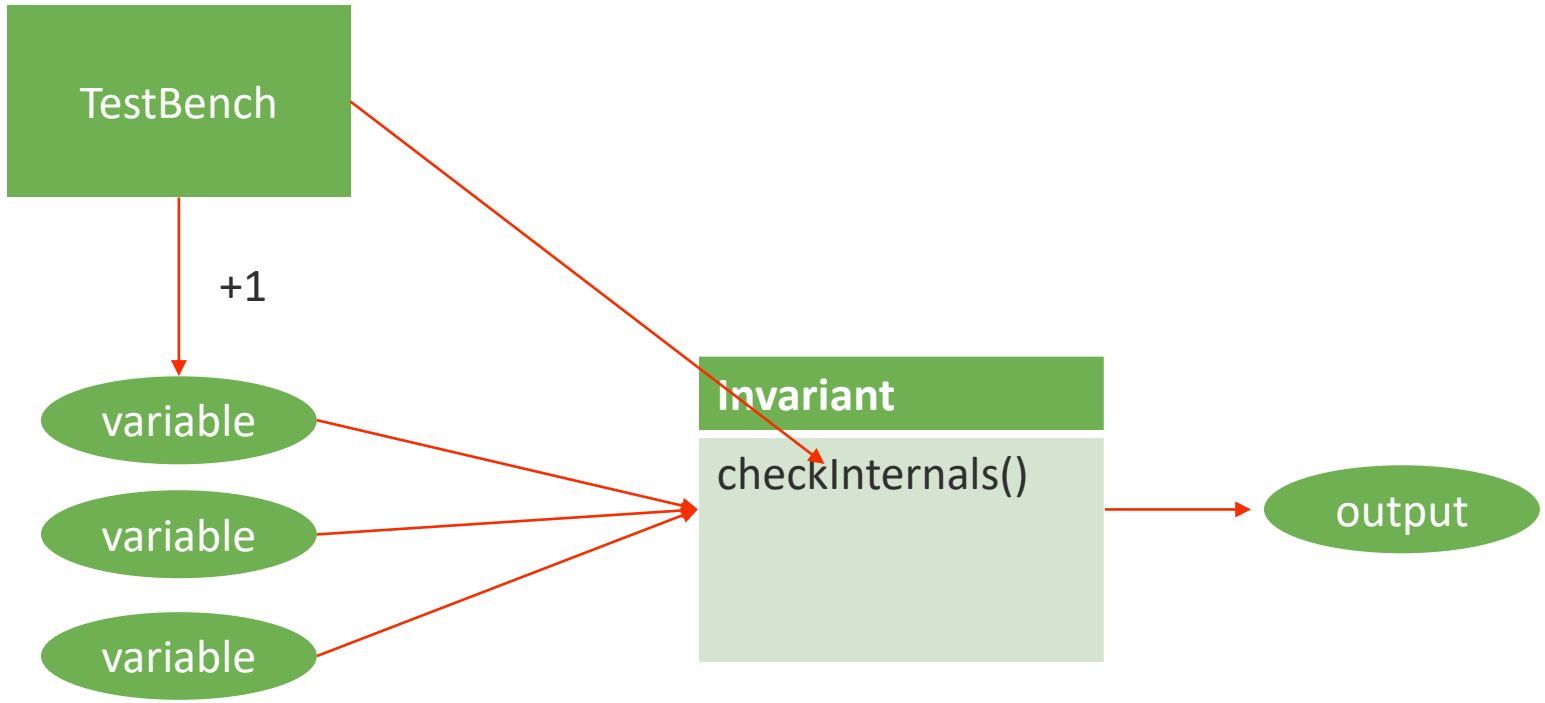
vehicle	length
0	20
1	32
2	5
...	...

Préambule : OscaR



Existant : TestBench

- **TestBench** : test des invariants
- Permet la rédaction d'un test en moins de 10 lignes
- Requiert que l'invariant implémente certaines interfaces

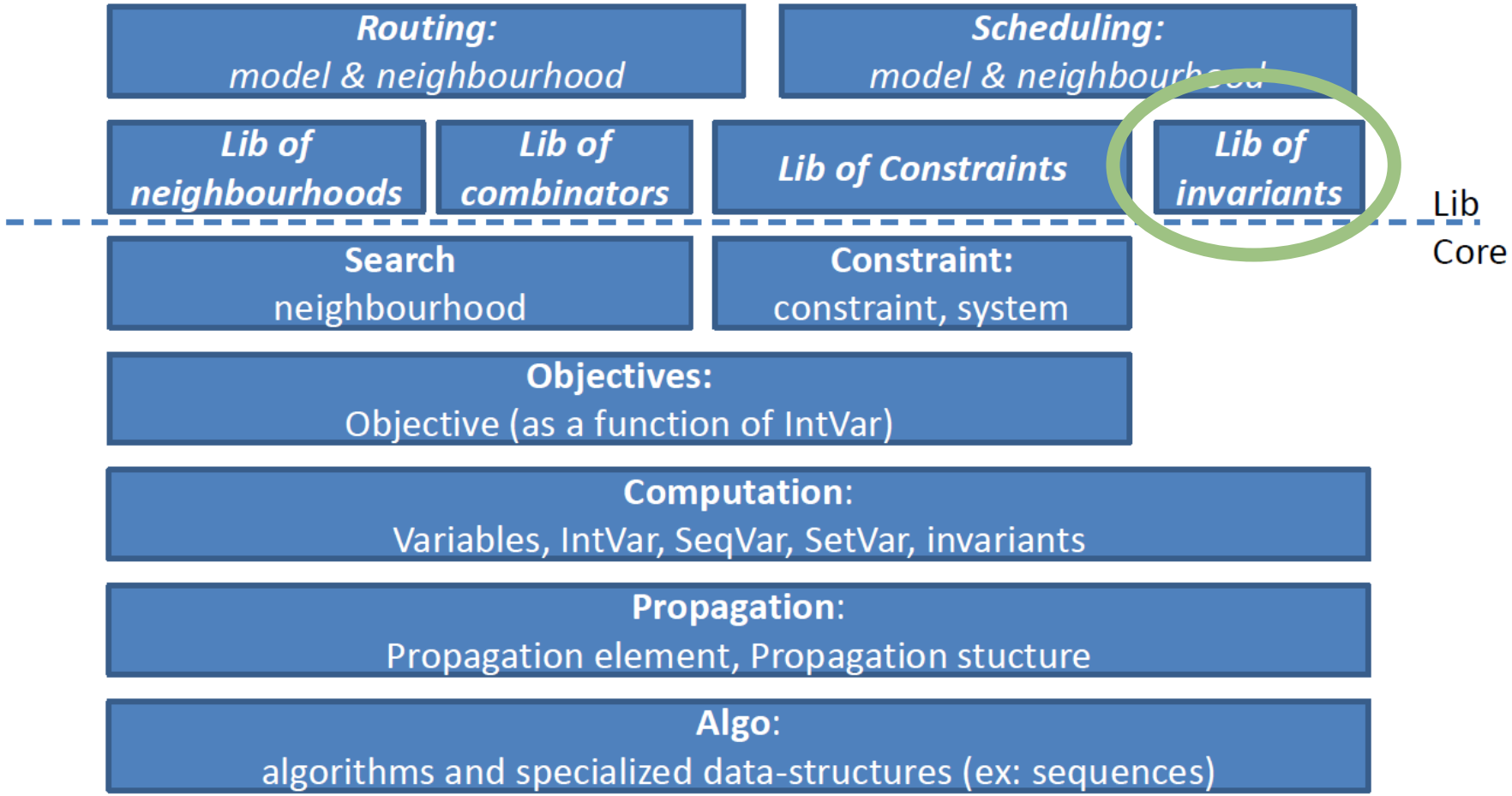


Existant : TestBench

- Exemple

```
test ("PositionsOf maintains the positions of a value"){
  val bench = new InvBench(verbose,
      List(PlusOne(), MinusOne(), ToZero(), ToMin(), ToMax(), Random()))
  val seqVar = bench.genIntSeqVar(range = 0 to 100)
  val value = bench.genIntVar(0 to 100)
  PositionsOf(seqVar,value)
  bench.run()
}
```

Scope du TestBench & Problématique



ScalaTest & ScalaCheck au secours

- **ScalaTest** : Format des assertions

Assertion standard

```
assert(elements.forall(myList.contains(_)),
       ‘All the elements were not in myList’ )
```

Assertion ‘scalatest’

```
myList should contain allElementsOf (elements)
```

Assertion standard

```
try{…} catch { case _ => caught = true}
assert(caught, ‘The exception wasn’t thrown’)
```

Assertion ‘scalatest’

```
an [Exception] should be thrownBy myMethodDestinedToFail()
```

ScalaTest & ScalaCheck au secours

- **ScalaTest** : Format des stacktraces

```
List(0, 1, 2, 3, 4, 5) did not contain all elements of List(8, 8)
ScalaTestFailureLocation: oscar.cbls.test.algo.FibonacciHeapTestSuite at
(FibonacciHeapTestSuite.scala:162)
org.scalatest.exceptions.TestFailedException: List(0, 1, 2, 3, 4, 5) did not contain element List(8, 8)
        at org.scalatest.MatchersHelper$.indicateFailure(MatchersHelper.scala:340)
        at org.scalatest.Matchers$ShouldMethodHelper$.shouldMatcher(Matchers.scala:6668)
        at org.scalatest.Matchers$AnyShouldWrapper.should(Matchers.scala:6716)
        at oscar.cbls.test.algo.FibonacciHeapTestSuite.$anonfun$new$15(FibonacciHeapTestSuite.scala:162)
```

```
Expected exception java.lang.Exception to be thrown, but no exception was thrown
ScalaTestFailureLocation: oscar.cbls.test.algo.FibonacciHeapTestSuite at
(FibonacciHeapTestSuite.scala:166)
```

ScalaTest & ScalaCheck au secours

- **ScalaCheck** : *Generator-driven & Property-based testing*

  - Le test unitaire ne spécifie pas d'input
  - Vérification de la validité d'une **propriété**
  - Structure typique :

    - Fonction F
    - Input arbitraire
    - F(input) doit satisfaire la condition C

```
test('Property-based example'){
    forAll(n: Int) =>

        val res = powerOfTwo(n)
        res should be >= n
        res should be (n*n)
}
```

Force 1 de Scalacheck : les données arbitraires

- Exploration plus efficace des inputs
- Messages clairs

```
! Falsified after 11 passed tests: > ARG_0 = "List(8, 0, -1, -3, -8, 8, 2, -10, 9, 1, -8)"
```

- Forcer un certain nombre d'itérations

```
test('Property-based example'){
    forAll(myGen,minSuccessful(1000)) =>

        ...
}
```

Force 2 de Scalacheck : les Generators

- Problème : la fonction à tester ne prends que des listes d'entiers positifs

```
test('Property-based example'){
    forAll((list: List[Int]) => {

        val res = allSquareRoots(list)
        res should be list.map(sqrt(_))
})
```

- Solution : Utiliser un Generator

```
val myGenerator = for{
  nbElems <- Gen.choose(1,1000)
  list <- Gen.listOfN(nbElems,Gen.choose(1,50))
} yield list
```

```
test('Generator-driven example'){
    forAll(myGenerator) list =>

        val res = allSquareRoots(list)
        res should be list.map(sqrt(_))
}
```

Force 3 de Scalacheck : le Shrink

- Fonction : identifier le plus petit input qui fait échouer le test
  - Exemple :

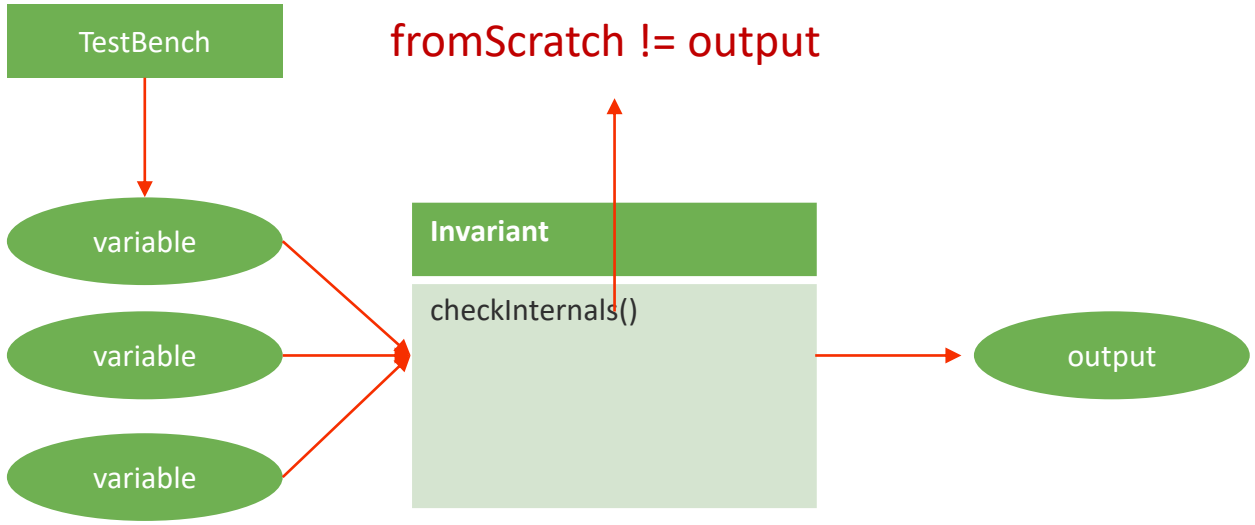
```
test("No duplicates"){
  forAll((lst :List[Int]) => {
    lst.sorted should be(lst.distinct.sorted)
  })
}
```

```
TestFailedException was thrown during property evaluation.
  Message: List(-1, -1) was not equal to List(-1)
  Location: (FibonacciHeapTestSuite.scala:13)
  Occurred when passed generated values (
    arg0 = List(-1, -1) // 3 shrinks
  )
```

Possibilité d'utiliser le Shrink en respectant les contraintes d'un Generator …

De retour au TestBench

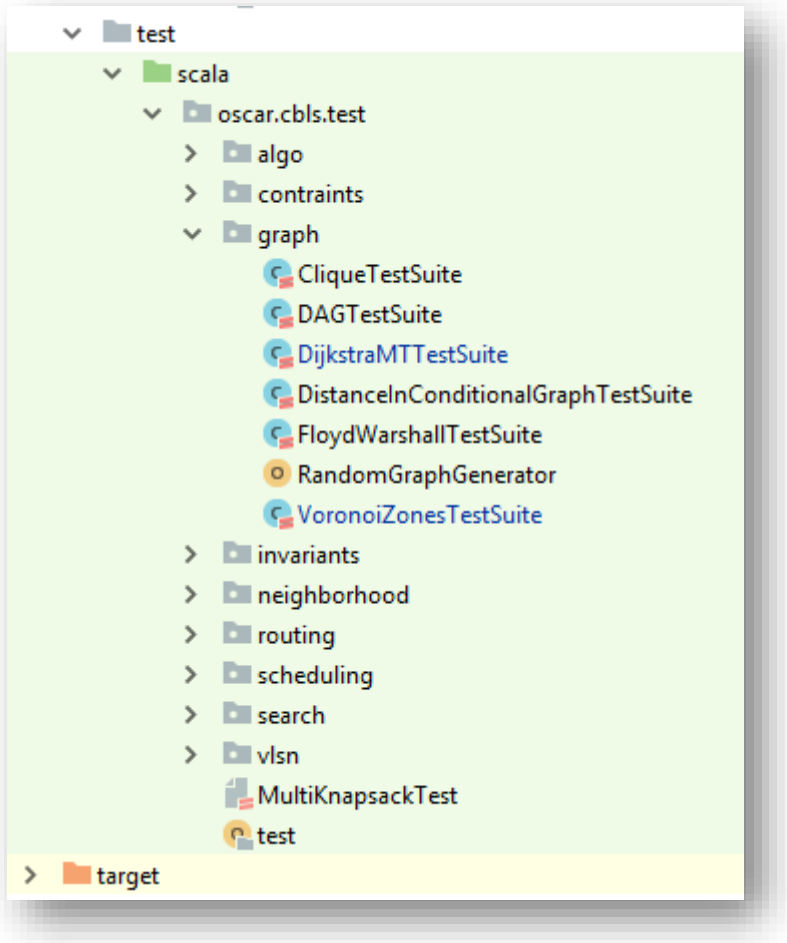
- Utilisation des forces de ScalaCheck pour le TestBench




```
Test failed during propagation, the internal error was :
    =>requirement failed: removing at position -1 size is 0
Occured after executing move PlusOne()
On variable class oscar.cbls.test.invariants.bench.RandomIntSeqVar
Variable before move B:=ConcreteIntSequence(size:3)[6,8,27]_impl:concrete
Variable after move B:=IntSequence(size:4)[6,69,8,27]_impl:[6,8,27]_impl:concrete.inserted(val:69 pos:1)
```

Mise en œuvre

- Suites de tests
  - Préférence pour les structures critiques
- Refactoring de tests existants
- Agrégation, organisation des fichiers



Couverture de code

Invariants

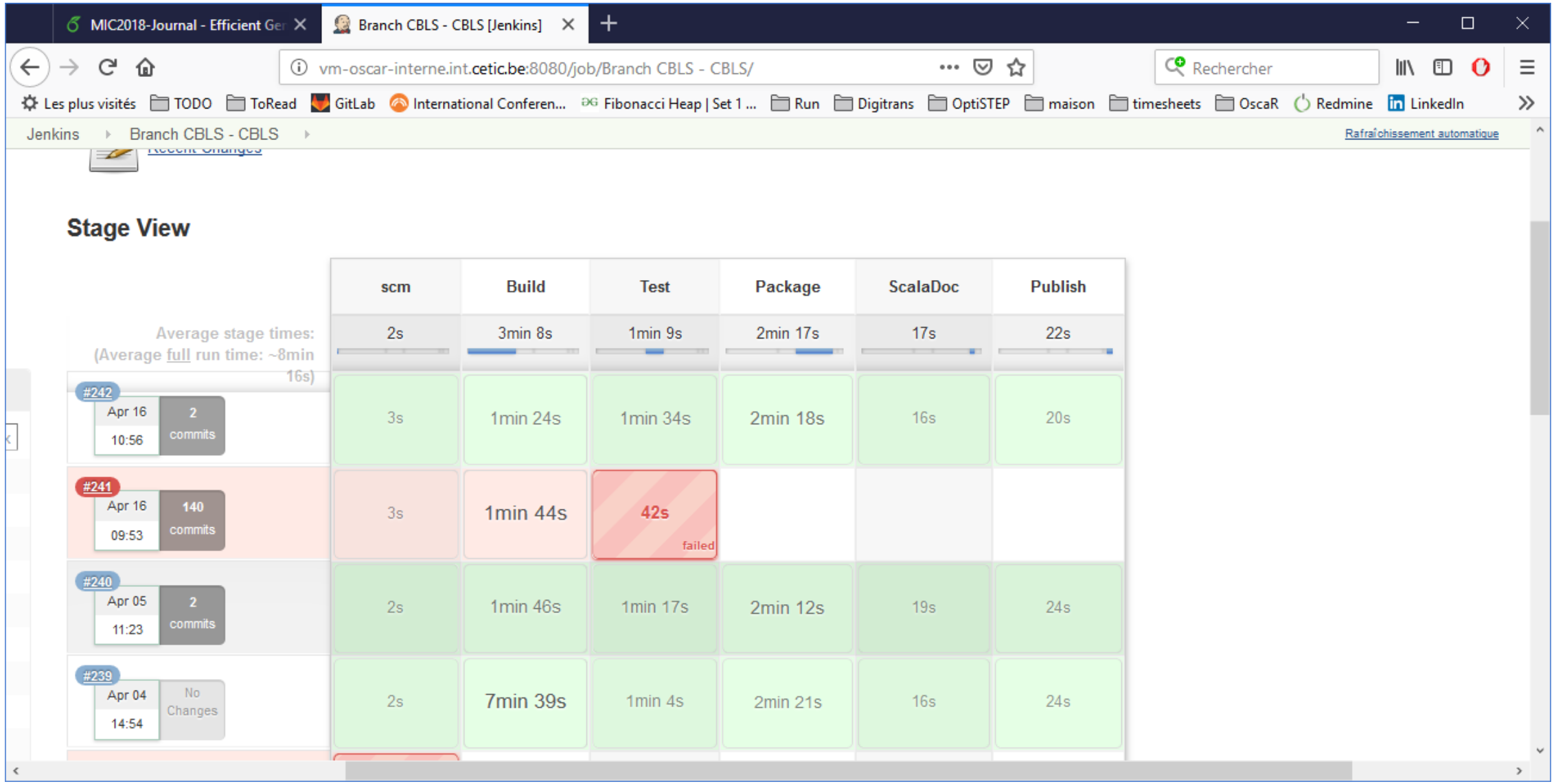
Element	Class, %	Method, %	Line, %
package	0% (0/1)	100% (0/0)	100% (0/0)
set	58% (20/34)	90% (138/153)	93% (332/355)
graph	66% (4/6)	84% (64/76)	88% (300/339)
minmax	76% (38/50)	77% (157/203)	75% (356/471)
seq	73% (25/34)	72% (135/187)	53% (511/949)
logic	26% (19/72)	50% (136/267)	48% (296/613)
numeric	33% (34/102)	38% (90/236)	30% (185/599)
package$	0% (0/1)	0% (0/1)	0% (0/2)

Algo

Element	Class, %	Method, %	Line, %
clique	50% (1/2)	100% (9/9)	100% (19/19)
package	0% (0/1)	100% (0/0)	100% (0/0)
seq	100% (20/20)	93% (164/175)	96% (623/645)
dag	100% (5/5)	91% (45/49)	93% (128/137)
dll	100% (7/7)	91% (51/56)	93% (100/107)
heap	92% (13/14)	78% (156/200)	88% (518/585)
magicArray	87% (7/8)	84% (50/59)	86% (112/130)
fun	84% (11/13)	69% (59/85)	79% (216/272)
graph	60% (14/23)	70% (73/104)	76% (250/326)
rb	39% (9/23)	55% (88/158)	62% (236/375)
search	35% (10/28)	42% (39/92)	42% (96/228)
distributedStora...	100% (2/2)	13% (2/15)	15% (4/26)
quick	14% (6/41)	15% (26/165)	14% (61/427)
tarjan	0% (0/6)	0% (0/17)	0% (0/93)
set	0% (0/2)	0% (0/18)	0% (0/70)

- Ennemis du code coverage :
  - *Code branch* particulières
  - Code déprécié
  - Boilerplate
  - `@Specialized`
  - Case class
  - …

Intégration à Jenkins



Bilans

- **Technique**
  - Augmentation de la couverture
  - Réorganisation des tests
  - Insertion de Generators réutilisables
  - Bonus : contribution aux algorithmes (Fibonacci Heap)
  - **Question ouverte** : Tests de voisinage ? …

- **Personnel**
  - Prise en main d'un projet de grande envergure
  - Bases de la Recherche Opérationnelle
  - Connaissances en algorithmique

Remerciements

Merci à :

- Renaud
- Christophe
- Gustavo
- Fabien
- Thomas
- Benjamin
- Et tout le monde au CETIC !

Super stage !

ScalaTest & ScalaCheck au service d'OscaR.CBLS

Arthur Attout

Stagiaire au département Optimisation

0491/52.16.80

arthur.attout@outlook.com

Références

« How does ScalaCheck shrinking works ? », Sanjiv Sahayam
https://blog.ssanj.net/posts/2017-04-12-how-does-scalacheck-shrinking-work.html

« ScalaCheck User Guide »
https://github.com/rickynils/scalacheck/blob/master/doc/UserGuide.md

« ScalaTest User Guide »
http://www.scalatest.org/user_guide

cetic

Your Connection to ICT Research

Aéropole de Charleroi-Gosselies
Avenue Jean Mermoz 28
6041 Charleroi - Belgique

twitter.com/@CETIC
twitter.com/@CETIC_be

linkedin.com/company/cetic

info@cetic.be

+32 71 159 362

www.cetic.be