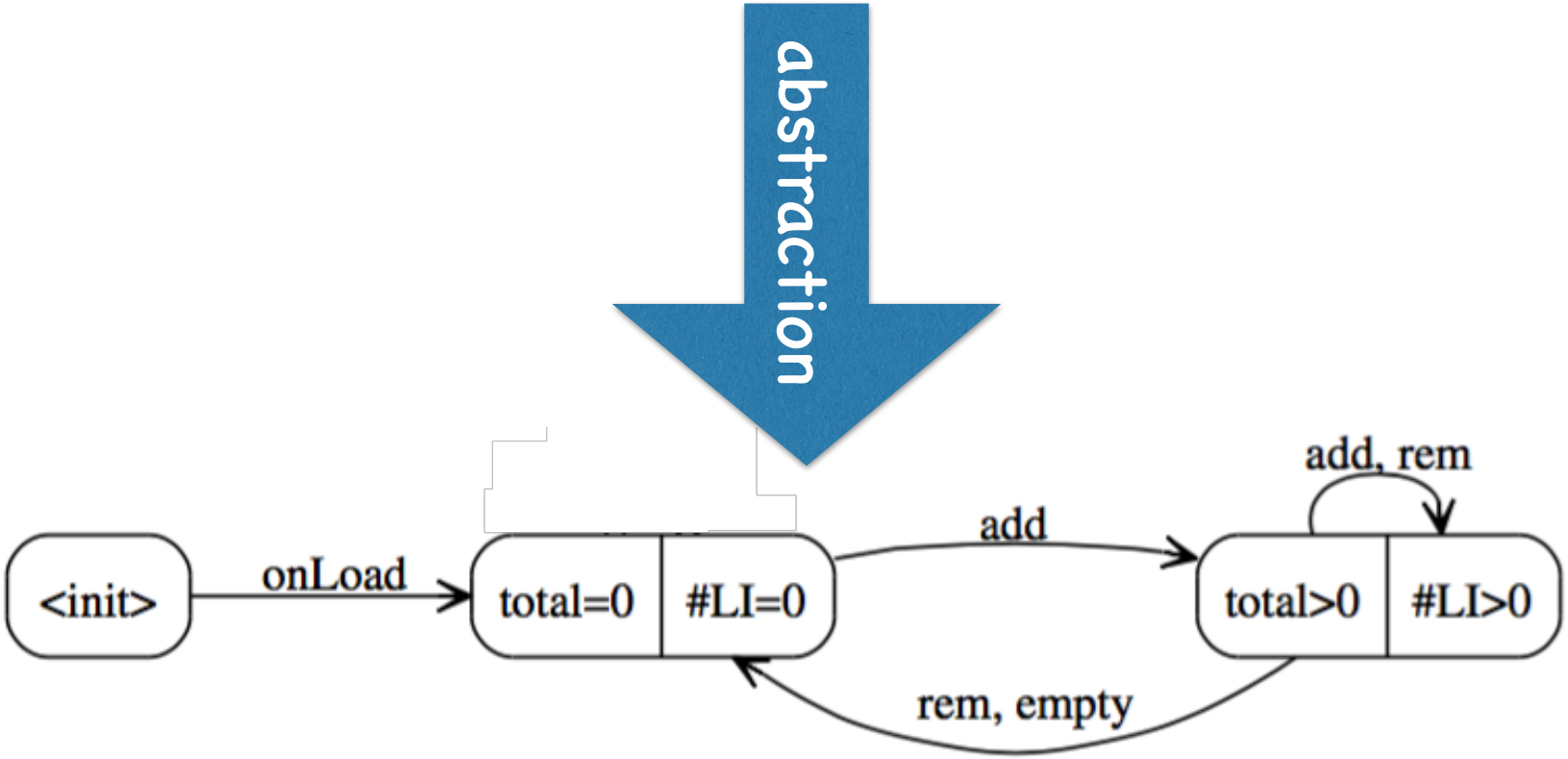
SuBWEB

Search based path and Input Data Generation for Web Application Testing

Matteo Biagiola, Paolo Tonella, Filippo Ricca

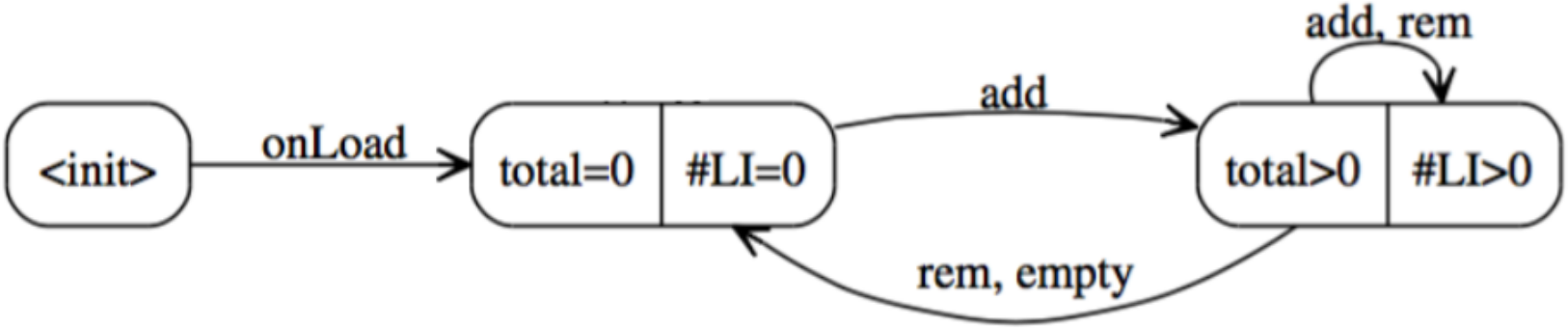
Why model-based testing?

- multi-tier architecture
- asynchronism
- different technologies



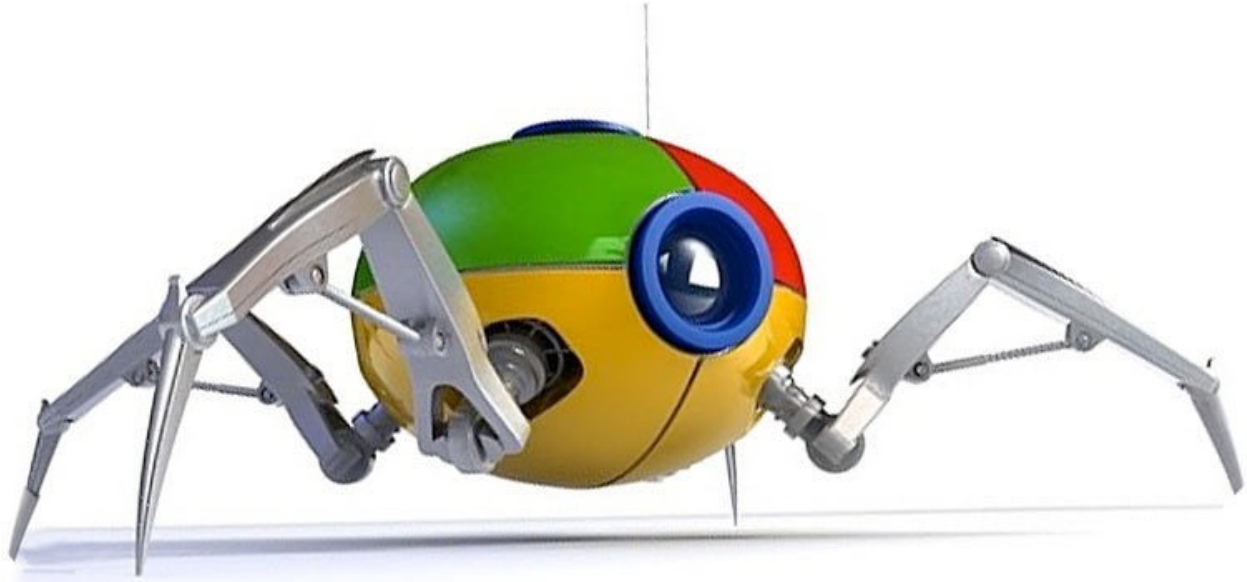
Problem definition

1) Getting the model
   - manual creation
   - automatic extraction



2) Using the model
   - test sequences generation
   - test data generation
   - test case feasibility

Existing approaches



1) Getting the model
   - automatic creation: web crawling based
2) Using the model
   - test sequences generation: graph visit algorithms
   - test data generation: manual or random
   - test case feasibility: not considered

Our approach

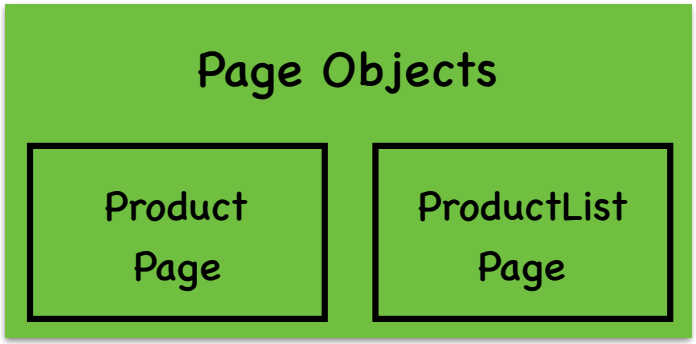
SuBWEB
(Search Based WEB test generator)

1) Getting the model
   - extraction from Page Objects
2) Using the model
   - joint generation of feasible test sequences and test data

PO design pattern (Web Testing)

App API

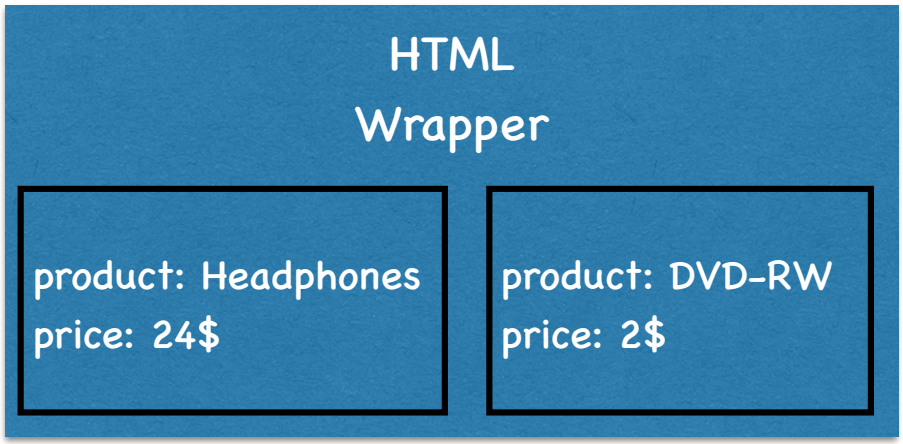
```
selectProductWithName()
getPrice()
updateProductPrice('52$')
```



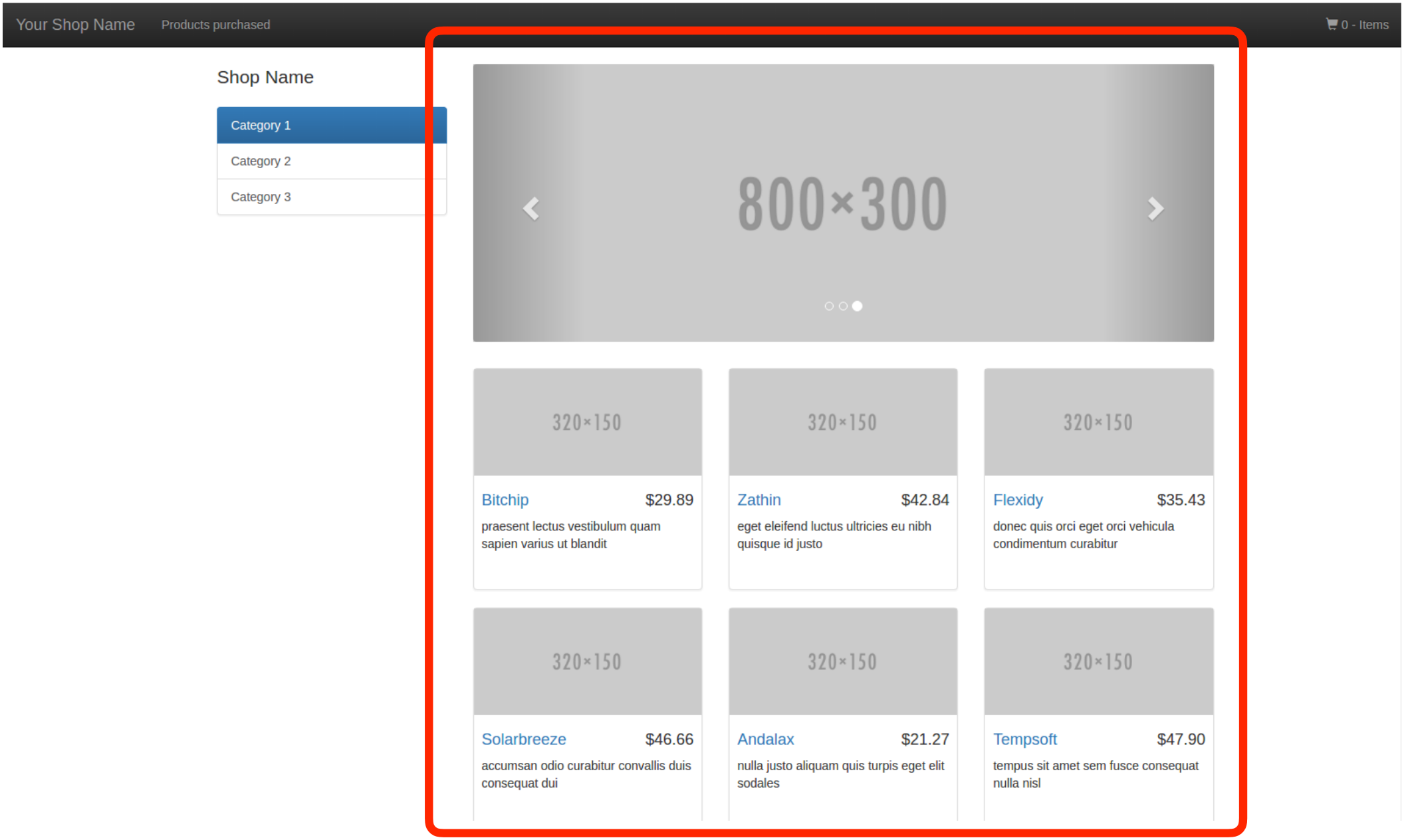
- reduce fragility
- increase reusability

HTML API

```
findElementsWithXpath('/html/div/h2/table/div[3]')
getText()
click()
setText('52$')
```



Shopping cart running example



Navigation model - specification via Page Objects

```
public class ProductsPage implements PageObject {
   public WebDriver driver;
   public ProductsPage(WebDriver driver) {...}
   public int getActiveCategory() {...}

   public ProductDetailPage selectProduct(int id, int category) {
      if((id >= 1 && id <= 6) &&
         (category >= 1 && category <= 3) &&
         (this.getActiveCategory() == category)) {
         this.driver.findElement(By.id("product-" + id + "-" +
             category)).click();
         return new ProductDetailPage(this.driver);
      } else {
         throw new IllegalArgumentException("Invalid parameter values");
      }
   }
}
```

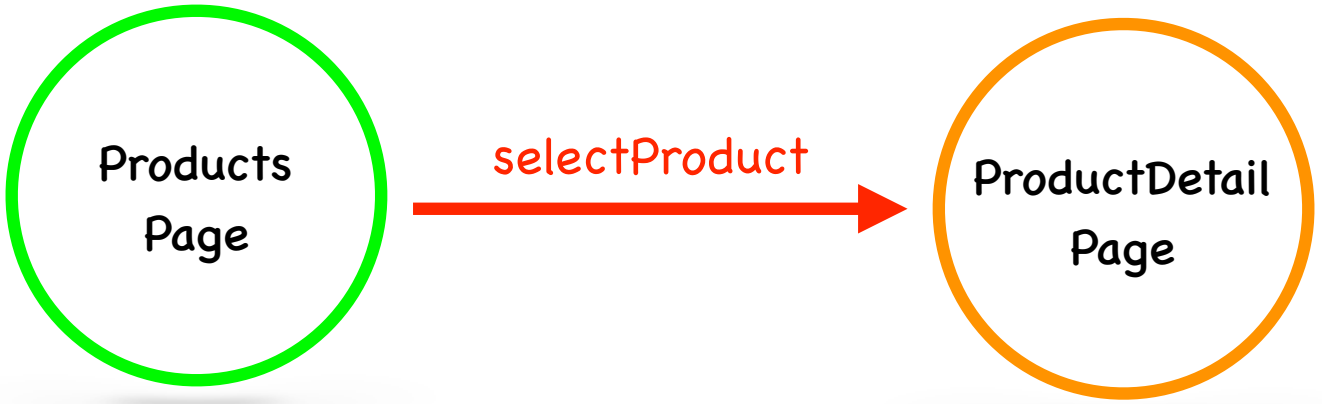
2nd assumption

1st assumption

Navigational model - automatic extraction

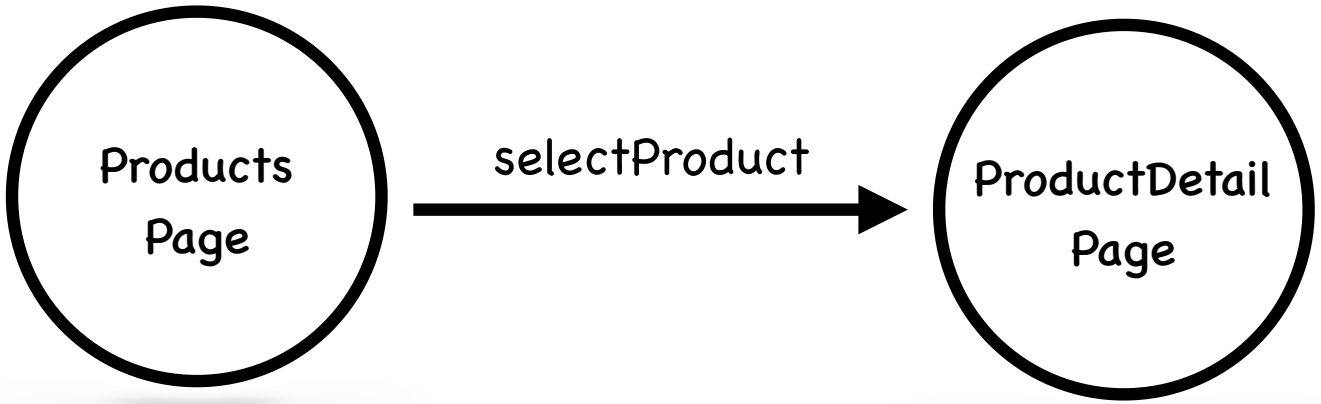
```
public class ProductsPage implements PageObject {
   public WebDriver driver;
   public ProductsPage(WebDriver driver) {...}
   public int getActiveCategory() {...}

   public ProductDetailPage selectProduct(int id, int category) {
      if((id >= 1 && id <= 6) &&
         (category >= 1 && category <= 3) &&
         (this.getActiveCategory() == category)) {
         this.driver.findElement(By.id("product-" + id + "-" +
             category)).click();
         return new ProductDetailPage(this.driver);
      } else {
         throw new IllegalArgumentException("Invalid parameter values");
      }
   }
}
```



Path

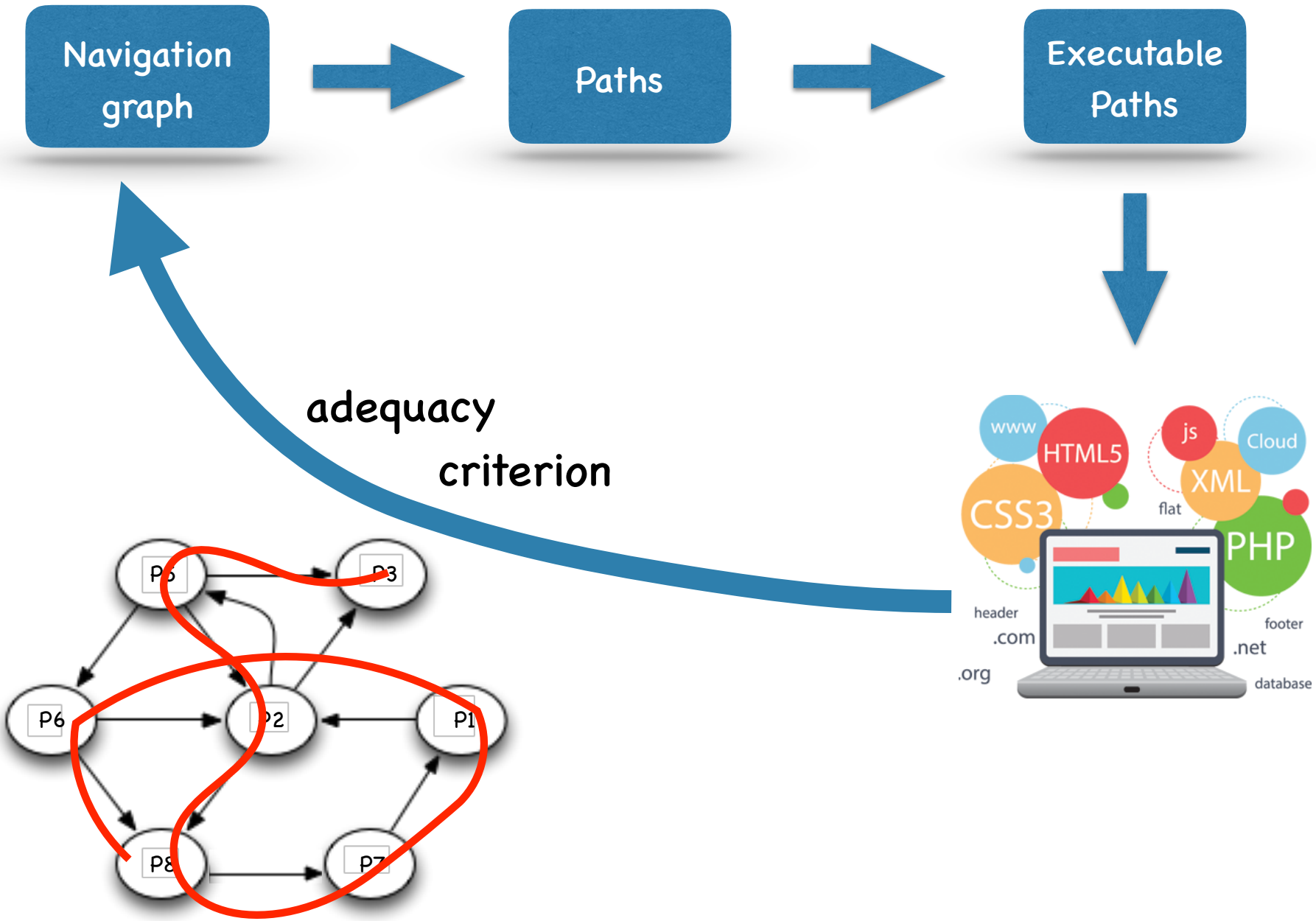
$$p = \langle ns, es, pr \rangle \quad ns \in N^+ \quad es \in E^* \quad pr \in V^*$$



$$ns = \langle \texttt{ProductsPage}, \texttt{ProductDetailPage} \rangle \quad pr = \langle \texttt{id}, \texttt{category} \rangle$$

$$es = \langle \texttt{selectProduct} \rangle$$

Test generation problem

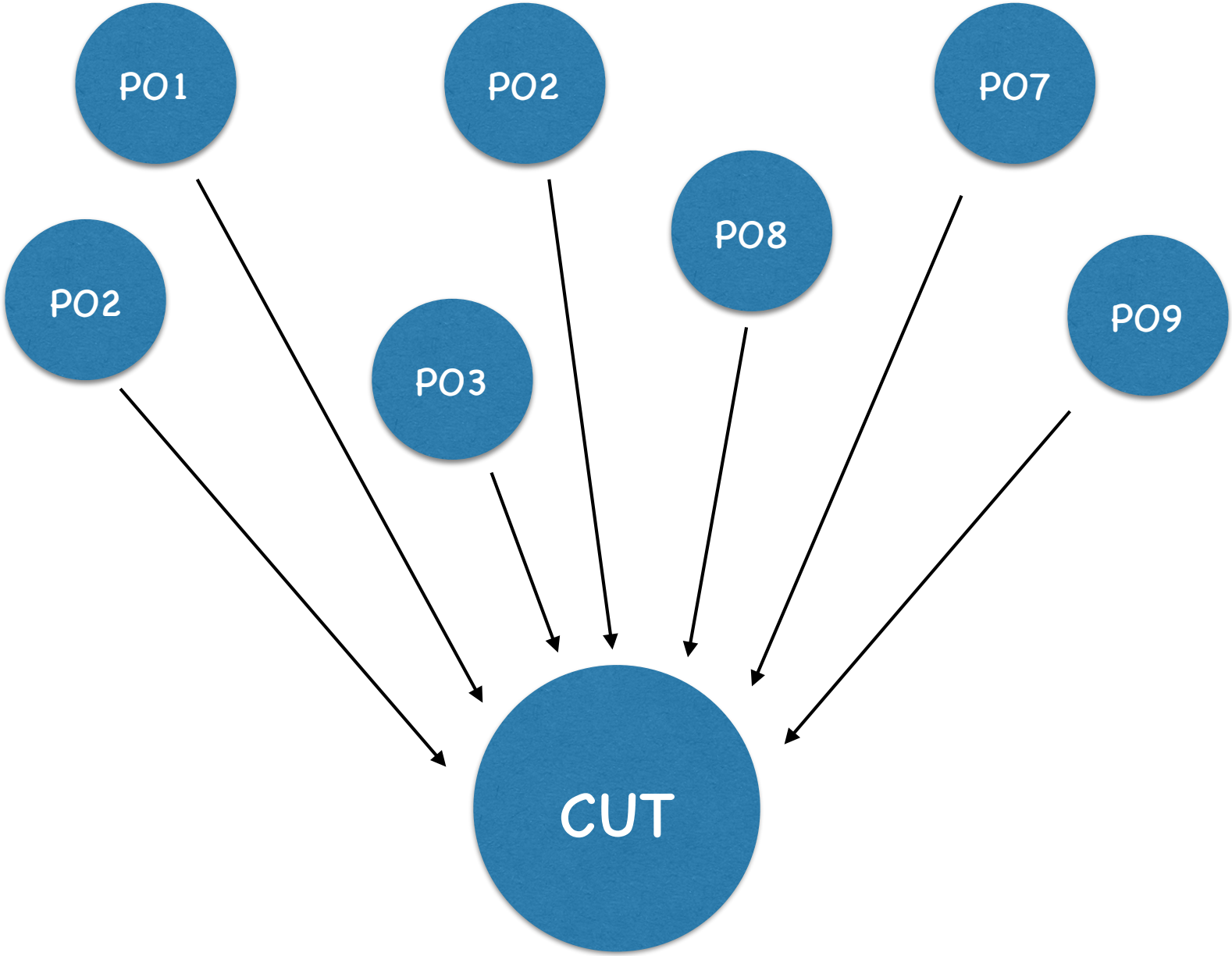


Test generation problem

Navigatio graph

utable ths

Problem reformulation - program transformation



Problem reformulation - program transformation

```
public class ProductsPage implements PageObject {
   public WebDriver driver;
   public ProductsPage(WebDriver driver) {...}
   public int getActiveCategory() {...}

   public ProductDetailPage selectProduct(int id, int category) {
      if((id >= 1 && id <= 6) &&
         (category >= 1 && category <= 3) &&
         (this.getActiveCategory() == category)) {
         this.driver.findElement(By.id("product-" + id + "-" +
             category)).click();
         return new ProductDetailPage(this.driver);
      } else {
         throw new IllegalArgumentException("Invalid parameter values");
      }
   }
}
```

Problem reformulation - program transformation

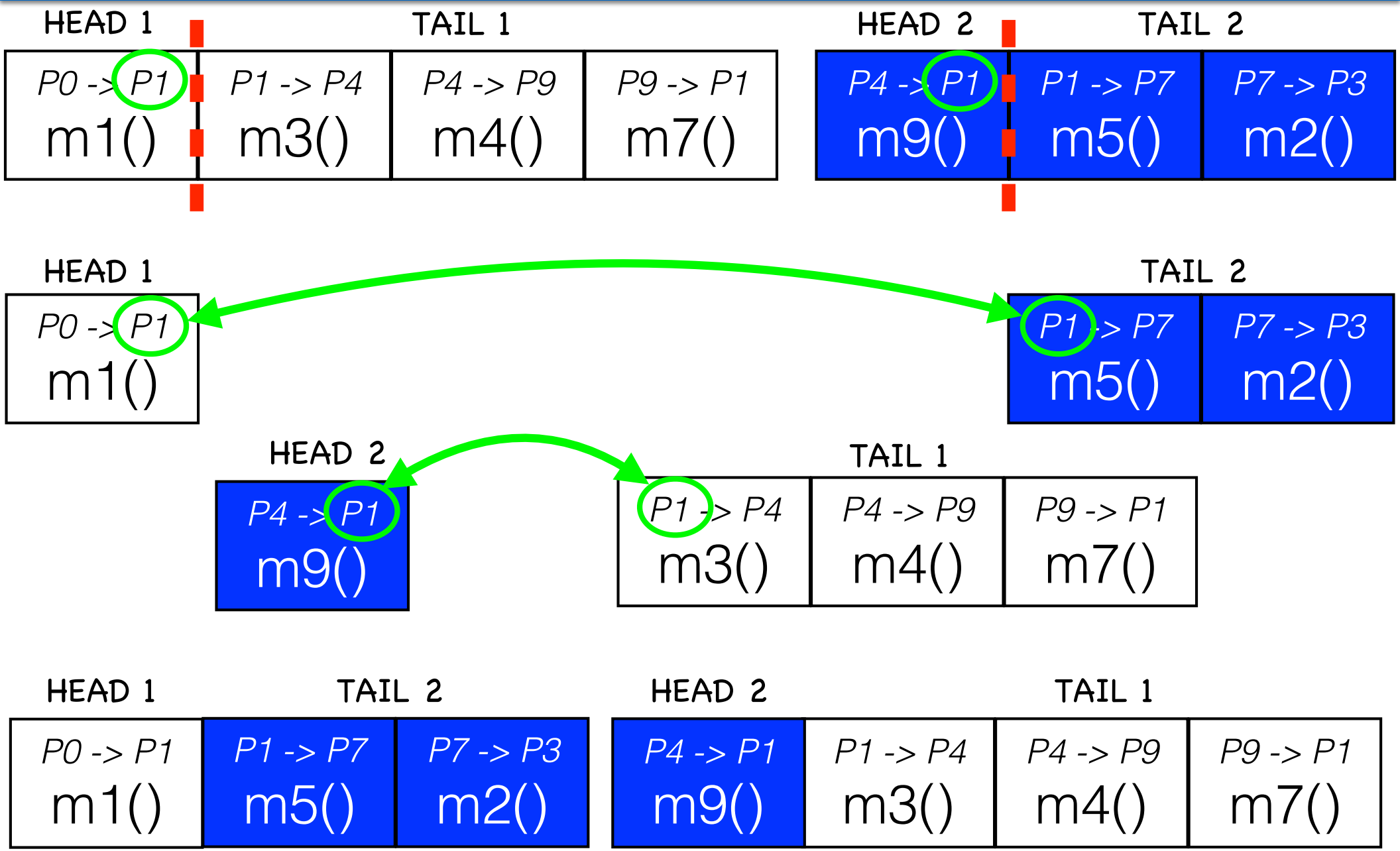
```
public class CUT {
   private PageObject currentPage;
   public void selectProduct(Id id, Category category) {
      if (this.currentPage instanceof ProductsPage) {
         ProductsPage page = (ProductsPage) this.currentPage;
         if(page.getActiveCategory() == category.value){
            page.driver.findElement(By.id("product-" + id.value + "-" + category.value)).click();
            this.currentPage = new ProductDetailPage(page.driver);
         } else { throw new IllegalArgumentException("Invalid parameter values"); }
      } else { throw new IllegalArgumentException("You are not in the right page"); }
   }
   ...
}
```

Problem reformulation - program transformation

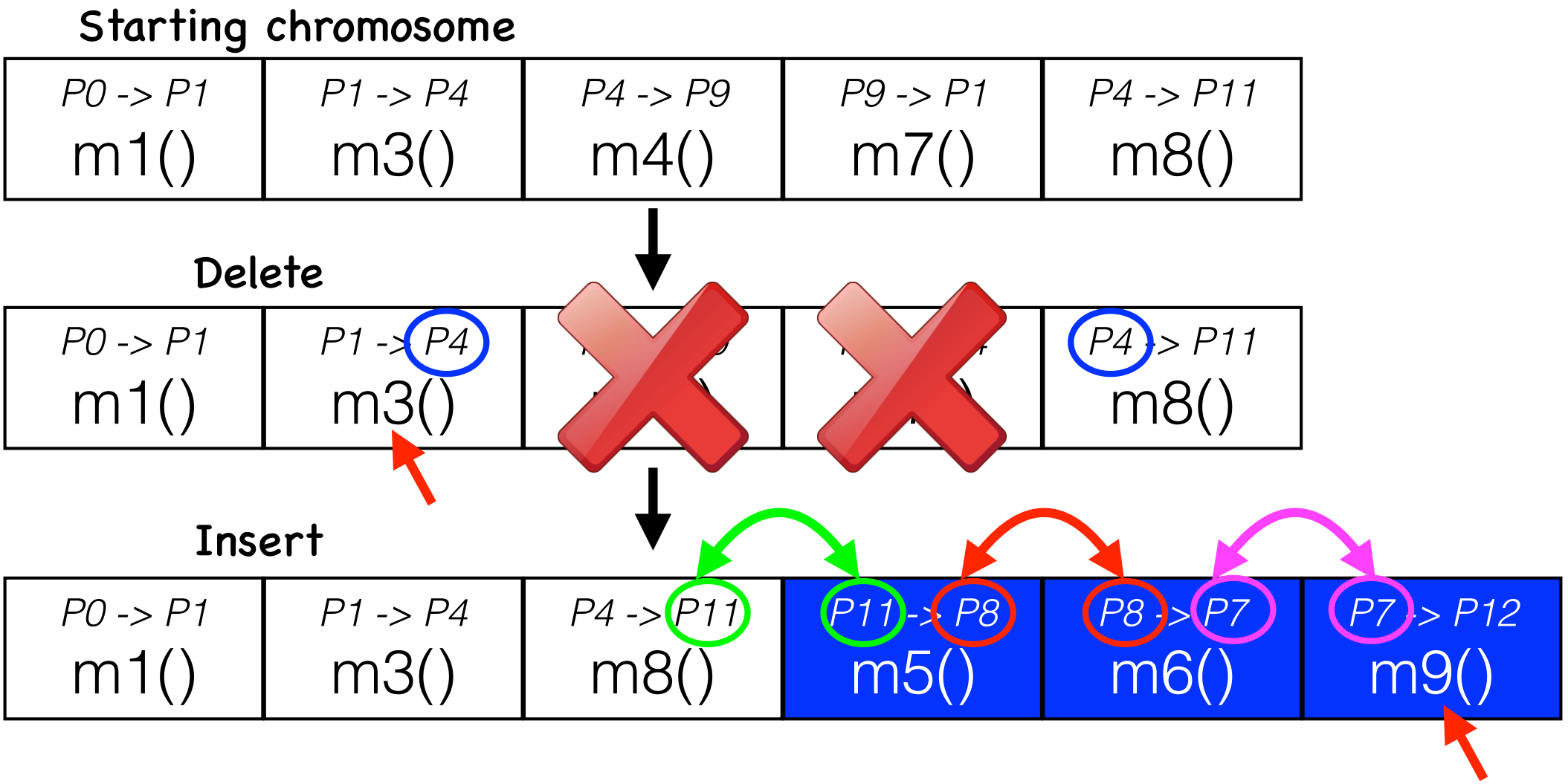
```
public class CU
   private PageO
   public void s
      if (this.c
         Product
         if(page
            page.
            this.
         } else
      } else { t
   }
   ...
}
```


```
alue)).click();
}
; }
```

Genetic operators - crossover targets matched



Genetic operators - mutation



Results - manual cost

App: AddressBook

PHP LOC	Javascript LOC
30223	1288

non trivial

POs

POs LOC	Number of POs	Navig. methods
764	1288	73

2%

Preconds

Methods LOC	Total number	Logic operators
75	16	54

0.2%

3

Results - models

PO graph

States	Transitions
12	73

Crawled graph

States	Transitions	Missing states	Missing transitions
329	927	0	5

27x states

13x transitions

Results - test suite size

SuBWEB

Test cases	Divergent test cases
54	0

Ext-Crawljax

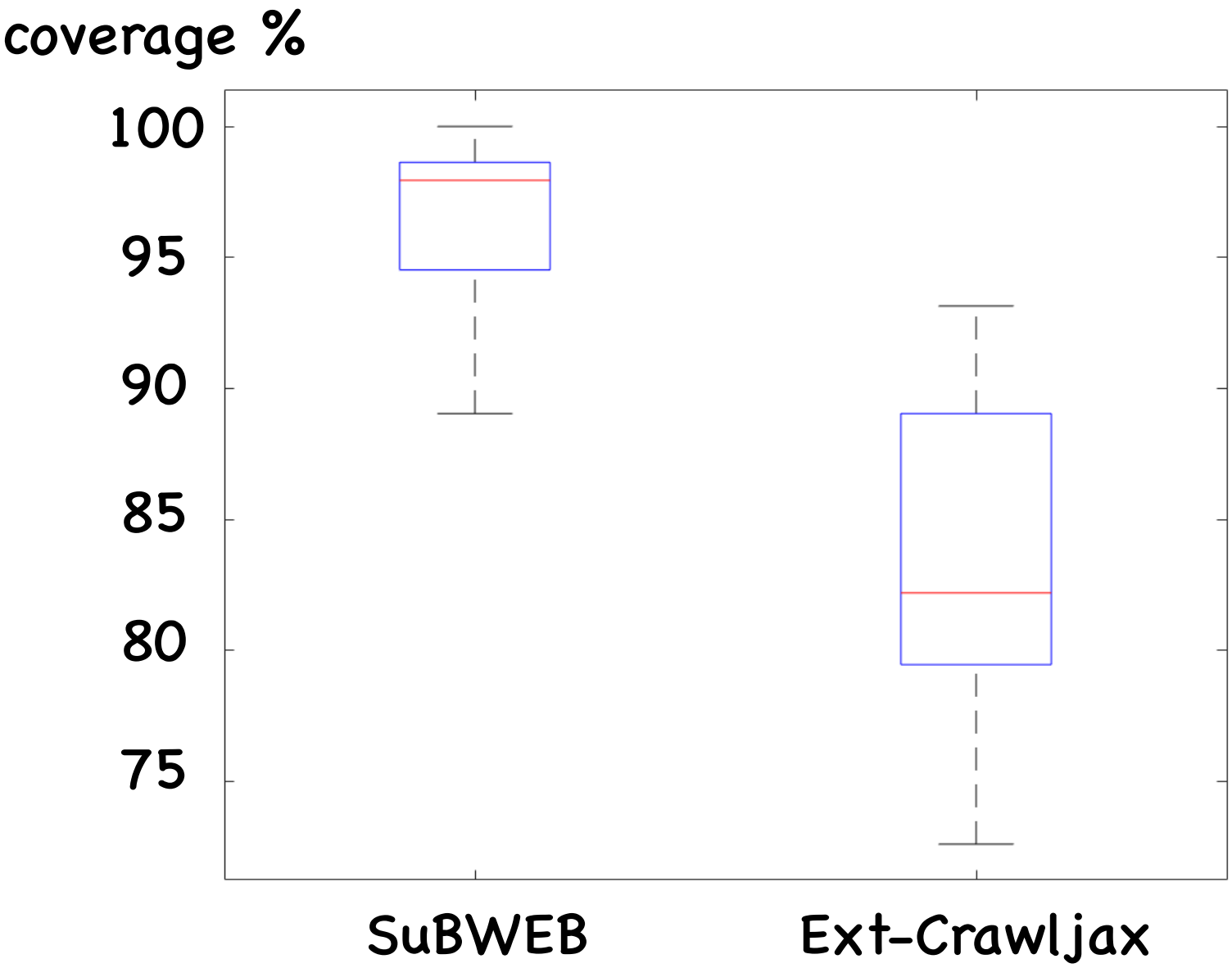
Test cases	Divergent test cases
598	104

11x size

17%

Divergence happens if an element existing at crawling time is no longer found at test time (different state): step in test case triggers an error so the desired path cannot be followed.

Results - coverage



Mann-Whitney U test

p-value
4.70×10^{-4}

Conclusions with comparisons

- Manual step

- Navigation graph is much smaller
- Smaller size test suites
- No divergency
- Higher transition coverage

Future works

Automatic generation of preconditions

Automatic generation of assertions

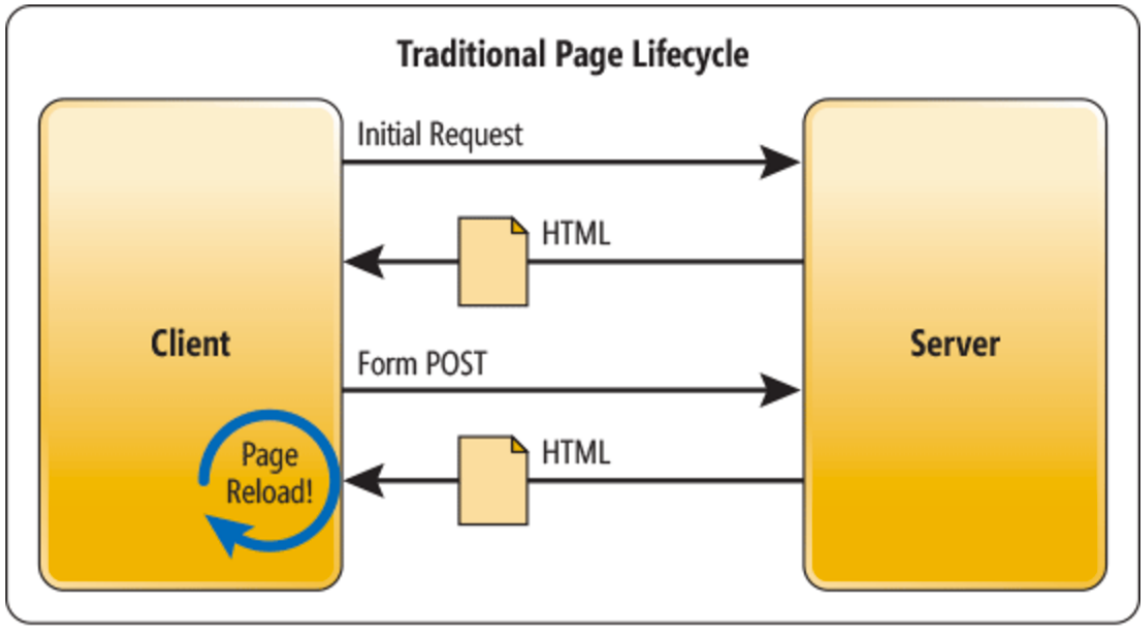
Thank you!

My supervisors

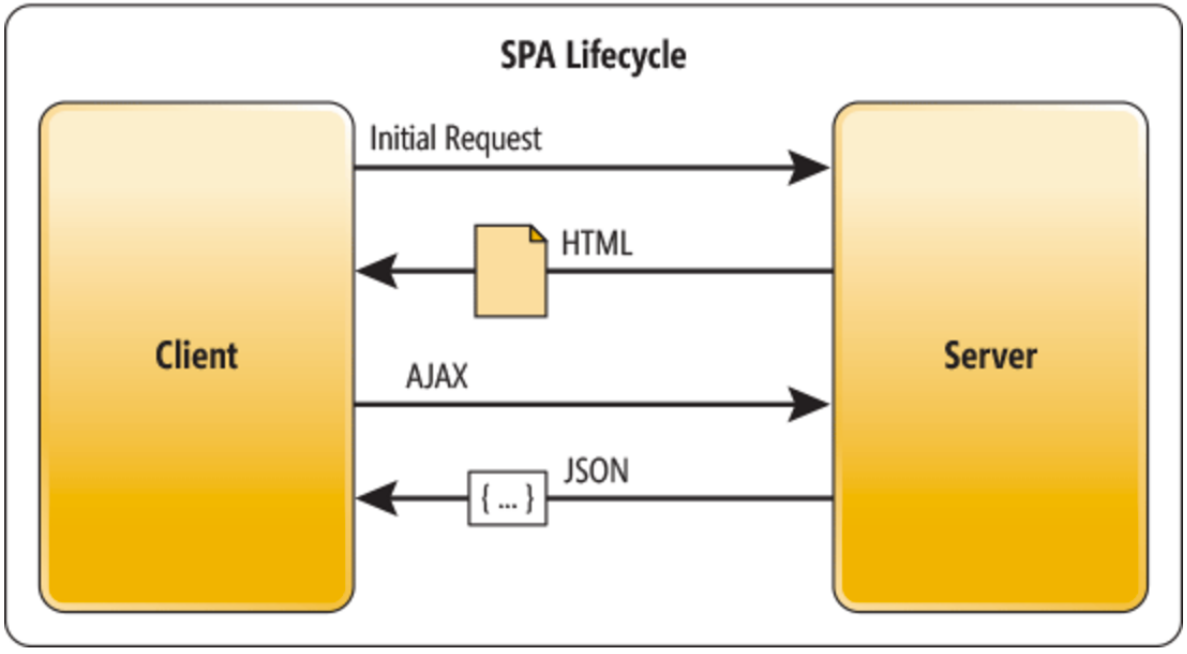
Paolo Tonella

Filippo Ricca

Web applications



- responsive
- separation client/server



Research questions

SuBWEB

RQ1

Manual cost

POs preconditions

SuBWEB VS Ext-Crawljax

RQ2

Navigation graph

States, transitions

RQ3

Test suite features

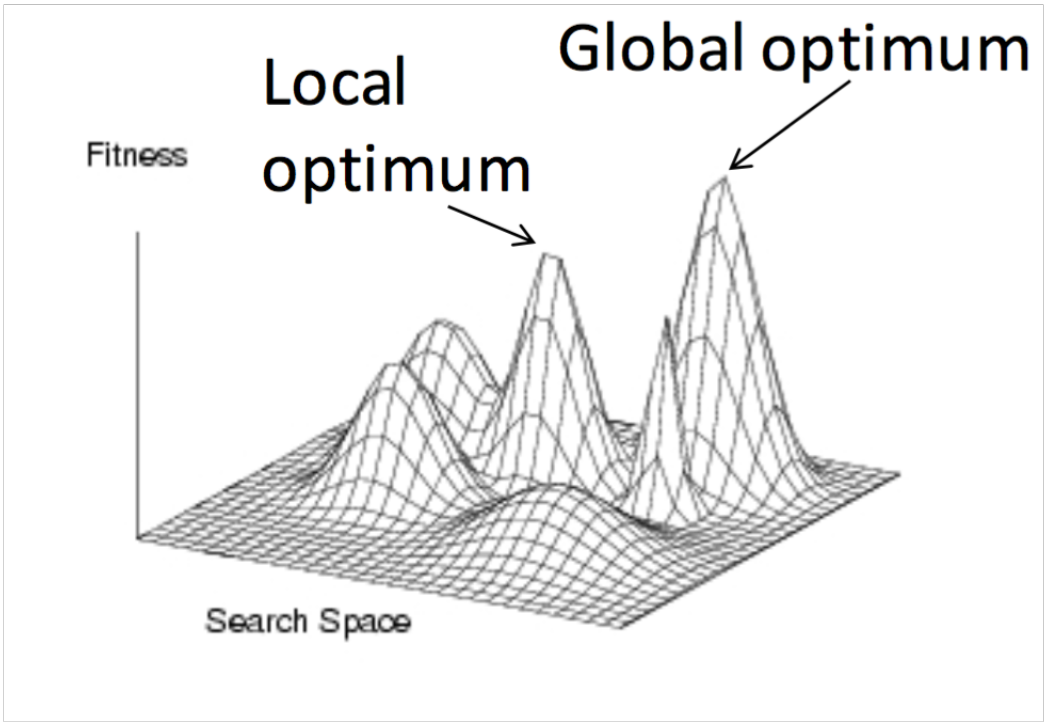
Size, % divergent

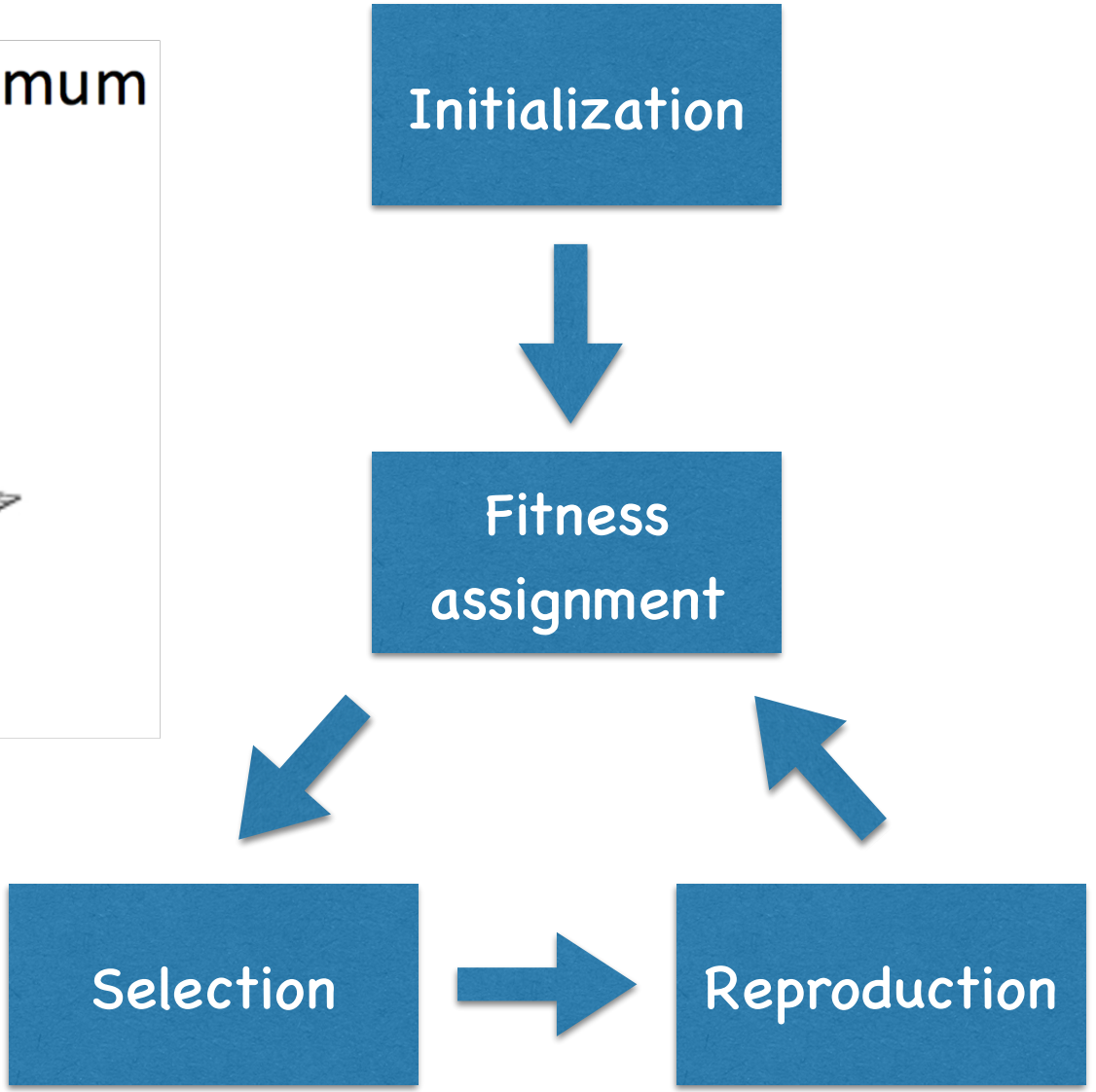
RQ4

Coverage

Navigation graph

Genetic algorithms

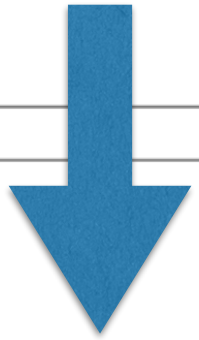




Problem reformulation - program transformation

```
public class ProductsPage implements PageObject {
   public WebDriver driver;
   public ProductsPage(WebDriver driver) {...}
   public int getActiveCategory() {...}

   public ProductDetailPage selectProduct(int id, int category) {
      if ((id >= 1 && id <= 6) &&
         (category >= 1 && category <= 3) &&
         (this.getActiveCategory() == category)) {
         this.driver.findElement(By.id("product-" + id + "-" +
             category)).click();
         return new ProductDetailPage(this.driver);
      } else {
         throw new IllegalArgumentException("Invalid parameter values");
      }
   }
}
```



```
public class CUT {
   private PageObject currentPage;
   public void selectProduct(Id id, Category category) {
     if (this.currentPage instanceof ProductsPage) {
        ProductsPage page = (ProductsPage) this.currentPage;
        if(page.getActiveCategory() == category.value){
          page.driver.findElement(By.id("product-" + id.value + "-" + category.value)).click();
          this.currentPage = new ProductDetailPage(page.driver);
        } else { throw new IllegalArgumentException("Invalid parameter values"); }
     } else { throw new IllegalArgumentException("You are not in the right page"); }
   }
}
```

Branch distance

Condition c = atomic predicate	Distance BD(c) = d / (d + 1)
a	d = {0 if a == true; K otherwise}
!a	d = {K if a == true; 0 otherwise}
a == b	d = {0 if a == b; abs(a - b) + K otherwise}
a != b	d = {0 if a != b; K otherwise}
a < b	d = {0 if a < b; a - b + K otherwise}
a <= b	d = {0 if a <= b; a - b + K otherwise}
a > b	d = {0 if a > b; b - a + K otherwise}
a >= b	d = {0 if a >= b; b - a + K otherwise}

Genetic algorithm - chromosome and fitness

chromosome

```
CUT cut0 = new CUT();
cut0.goToShoppingCartProductsPage();
cut0.goToShoppingCartShoppingCartPage();
cut0.goToHomeShoppingCartPage();
cut0.goToProductsPurchasedProductsPage();
cut0.goToHomeProductsPurchasedPage();
int int0 = (-2140);
Id id0 = new Id(int0);
Category category0 = new Category(id0.value);
cut0.selectProductProductsPage(id0, category0);
cut0.goToProductsPurchasedProductDetailPage();
cut0.goToHomeProductsPurchasedPage();
cut0.goToProductsPurchasedProductsPage();
cut0.continueShoppingProductsPurchasedPage();
```

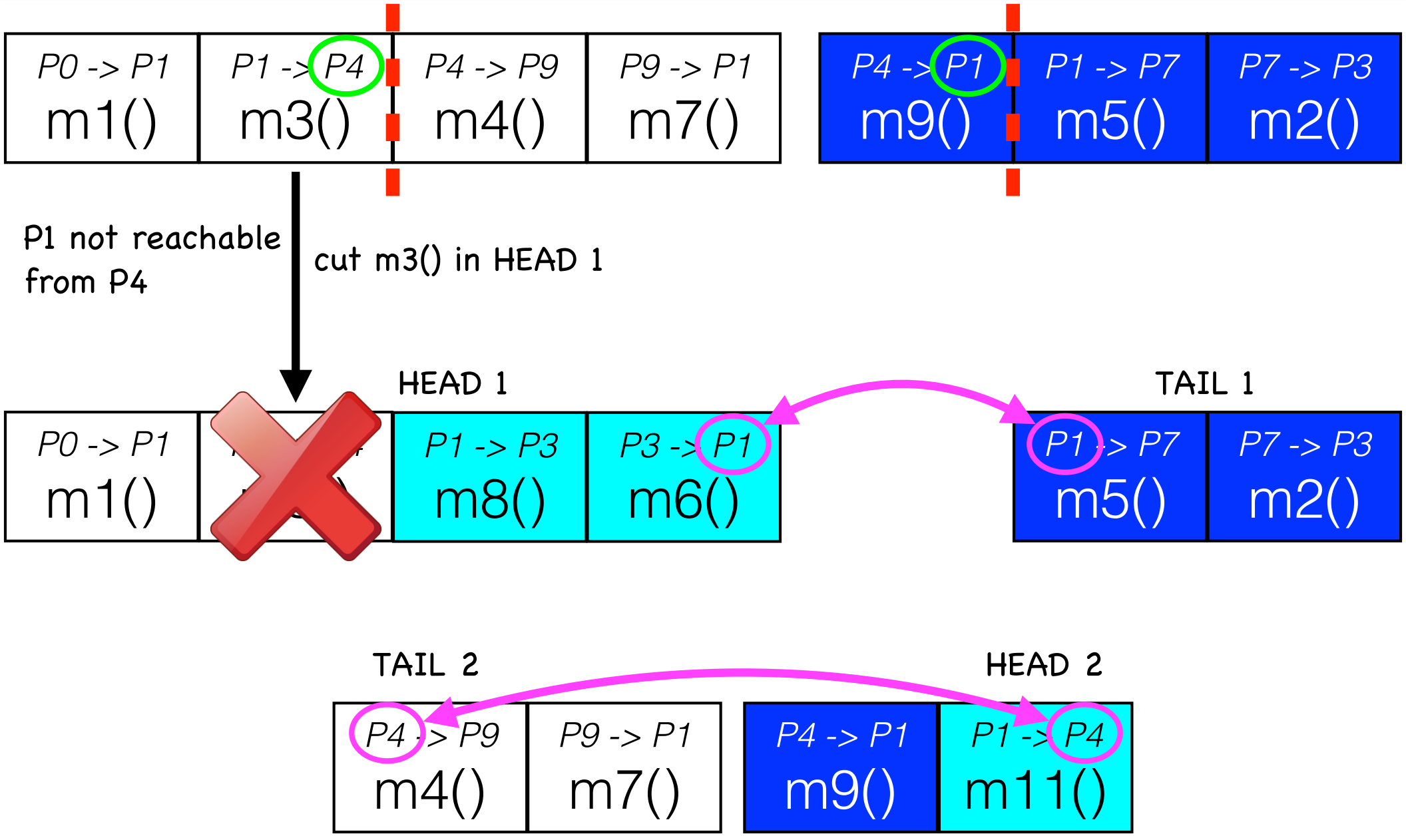
branch coverage criterion

$$\text{fitness}(T) = |M| - |M_T| + \sum_{b_k \in B} d(b_k, T)$$

unexecuted methods

uncovered branches

Genetic operators – crossover targets do not match



Genetic operators - mutation insertion non reachability