

Playing with Refactoring

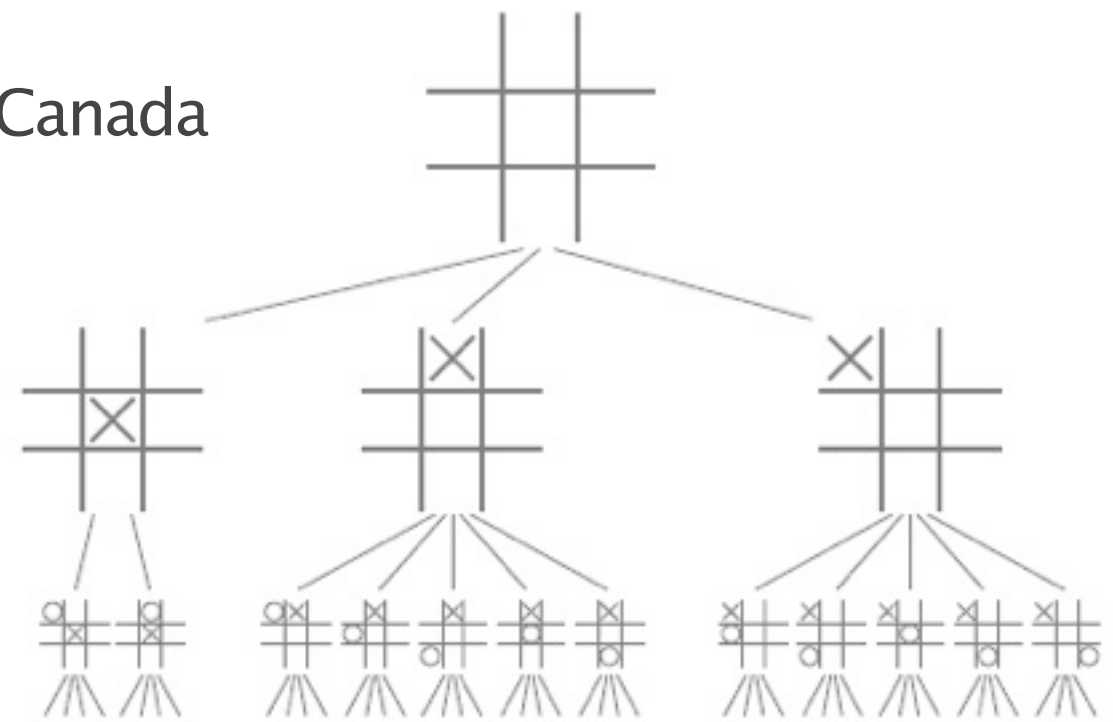
Identifying Extract Class Opportunities through Game Theory

Gabriele Bavota^{*}, Rocco Oliveto^{*}, Andrea De Lucia^{*}

Giuliano Antoniol[†], Yann-Gaël Guéhéneuc[†]

^{*} DMI, University of Salerno, Fisciano (SA), Italy

[†] DGIGL, École Polytechnique de Montréal, Québec, Canada





Game Theory Background

The Prisoner's Dilemma



Context

Refactoring Software Systems: Why and How



Game Theory meets SE

Game-based Extract Class Refactoring

contents

Game Theory Background

A close-up photograph of a person's hands in metal handcuffs. The hands are holding a knife with a dark handle and a sharp blade. The lighting is dramatic, with strong highlights on the skin and the metal of the cuffs, set against a dark, blurred background.

The Prisoner's Dilemma

Game Theory



- is a branch of mathematics widely applied in the social sciences
- capture behavior in strategic situations, in which an individual's success in making choices depends on the choices of others
- *a game consists of:*



- a set of players (2 or more);
- a set of moves available to those players;
- payoffs for each combination of moves

The Prisoner's Dilemma

		Tom	
		confess	not confess
Sally	confess	(5, 5)	(0, 7)
	not confess	(7, 0)	(4, 4)

The Prisoner's Dilemma

		Tom	
		confess	not confess
Sally	confess	NASH EQUILIBRIUM (5, 5)	(0, 7)
	not confess	(7, 0)	(4, 4)

Game Theory: Summarizing



- Natural application in strategic situations
- How to find a compromise between contrasting goals (Nash Equilibrium)

- *In software engineering:*



- optimal solution to many problems involves finding a compromise between contrasting goals, e.g., create classes with high cohesion and low coupling

Context



Refactoring Software Systems: Why and How

Refactoring...Why?



- Changing software *without* modifying its external behaviour
- Improve non-functional attributes of the software



- Software evolution ... continuous changes
- Changes cause a drift of the original design, reducing its quality, e.g., Class Cohesion

Focusing on Class Cohesion



- How strongly related and focused the various responsibilities of a class are
- High cohesion is desirable...easier maintenance
- Programmers often add wrong responsibilities to a class
- The class becomes too complex and its cohesion decreases

class
class
class
class
class
class
class

Focusing on Class Cohesion



- How strongly related and focused the various responsibilities of a class are
- High cohesion is desirable...easier maintenance

class

Extract Class Refactoring

class class

Splitting a class with many responsibilities into different classes

Game Theory meets SE



Game-based Extract Class Refactoring

Playing with Refactoring

MODELLING A
NON-COOPERATIVE GAME

..choosing the better strategy

		<div><div>T</div><div>m₅</div></div>			
		m ₂	m ₃	m ₄	N
<div><div>S</div><div>m₁</div></div>	m ₂	(-1.00, -1.00)	(0.49, 0.22)	(0.70, 0.80)	(0.70, 0.50)
	m ₃	(-0.49, -0.24)	(-1.00, -1.00)	(0.21, 0.58)	(0.21, 0.28)
	m ₄	(-0.70, -0.80)	(-0.21, -0.58)	(-1.00, -1.00)	(0.00, -0.30)
	N	(-0.20, 0.00)	(0.29, 0.22)	(0.50, 0.80)	(-1.00, -1.00)

Playing with Refactoring

MODELLING A
NON-COOPERATIVE GAME

..choosing the better strategy

$(-1, -1)$ if $i = j$

T

m_5

m_2

m_3

m_4

N

S

m_2

$(-1.00, -1.00)$ $(0.49, 0.22)$ $(0.70, 0.80)$ $(0.70, 0.50)$

m_3

$(-0.49, -0.24)$ $(-1.00, -1.00)$ $(0.21, 0.58)$ $(0.21, 0.28)$

m_4

$(-0.70, -0.80)$ $(-0.21, -0.58)$ $(-1.00, -1.00)$ $(0.00, -0.30)$

N

$(-0.20, 0.00)$ $(0.29, 0.22)$ $(0.50, 0.80)$ $(-1.00, -1.00)$

Playing with Refactoring

MODELLING A
NON-COOPERATIVE GAME

..choosing the better strategy

T

m₅

$$0.70 = \text{sim}(m_1, m_2) - \text{sim}(m_1, m_4)$$

m₂

m₃

m₄

N

S

m₁

m₂	(-1.00, -1.00)	(0.49, 0.22)	(0.70, 0.80)	(0.70, 0.50)
m₃	(-0.49, -0.24)	(-1.00, -1.00)	(0.21, 0.58)	(0.21, 0.28)
m₄	(-0.70, -0.80)	(-0.21, -0.58)	(-1.00, -1.00)	(0.00, -0.30)
N	(-0.20, 0.00)	(0.29, 0.22)	(0.50, 0.80)	(-1.00, -1.00)

Playing with Refactoring

MODELLING A
NON-COOPERATIVE GAME

..choosing the better strategy

T

m₅

$$0.70 = \text{COHESION} - \text{COUPLING}$$
$$0.70 = \text{sim}(m_1, m_2) - \text{sim}(m_1, m_4)$$

m₂

m₃

m₄

N

S

m₁

m₂	m₃	m₄	N
(-1.00, -1.00)	(0.49, 0.22)	(0.70, 0.80)	(0.70, 0.50)
(-0.49, -0.24)	(-1.00, -1.00)	(0.21, 0.58)	(0.21, 0.28)
(-0.70, -0.80)	(-0.21, -0.58)	(-1.00, -1.00)	(0.00, -0.30)
(-0.20, 0.00)	(0.29, 0.22)	(0.50, 0.80)	(-1.00, -1.00)

Playing with Refactoring

MODELLING A
NON-COOPERATIVE GAME

..choosing the better strategy

		<div><div>T</div><div>m₅</div></div>			
		m ₂	m ₃	m ₄	N
<div><div>S</div><div>m₁</div></div>	m ₂	(-1.00, -1.00)	(0.49, 0.22)	(0.70, 0.80)	(0.70, 0.50)
	m ₃	(-0.49, -0.24)	(-1.00, -1.00)	(0.21, 0.58)	(0.21, 0.28)
	m ₄	(-0.70, -0.80)	(-0.21, -0.58)	(-1.00, -1.00)	(0.00, -0.30)
	N	(-0.20, 0.00)	(0.29, 0.22)	(0.50, 0.80)	(-1.00, -1.00)

Playing with Refactoring

MODELLING A
NON-COOPERATIVE GAME

..choosing the better strategy

		<div><div>T</div><div>m₅</div></div>			
		m ₂	m ₃	m ₄	N
<div><div>S</div><div>m₁</div></div>	m ₂	(-1.00, -1.00)	(0.49, 0.22)	(0.70, 0.30)	(0.70, 0.50)
	m ₃	(-0.49, -0.24)	(-1.00, -1.00)	(0.21, 0.58)	(0.21, 0.28)
	m ₄	(-0.70, -0.80)	(-0.21, -0.58)	(-1.00, -1.00)	(0.00, -0.30)
	N	(-0.20, 0.00)	(0.29, 0.22)	(0.50, 0.80)	(-1.00, -1.00)

NASH
EQUILIBRIUM

Playing with Refactoring

MODELLING A
NON-COOPERATIVE GAME

..choosing the better strategy

		T				m ₅			
		m ₂	m ₃	m ₄	N				
S		(-1.00, -1.00)	(0.49, 0.22)	(0.70, 0.30)	(0.70, 0.50)	NASH EQUILIBRIUM			
	m ₃	(-0.49, -0.24)	(-1.00, -1.00)	(0.21, 0.58)	(0.21, 0.28)				
	m ₄	(-0.70, -0.80)	(-0.21, -0.58)	(-1.00, -1.00)	(0.00, -0.30)				
	N	(-0.20, 0.00)	(0.29, 0.22)	(0.50, 0.80)	(-1.00, -1.00)				

Playing with Refactoring

MODELLING A
NON-COOPERATIVE GAME

..choosing the better strategy

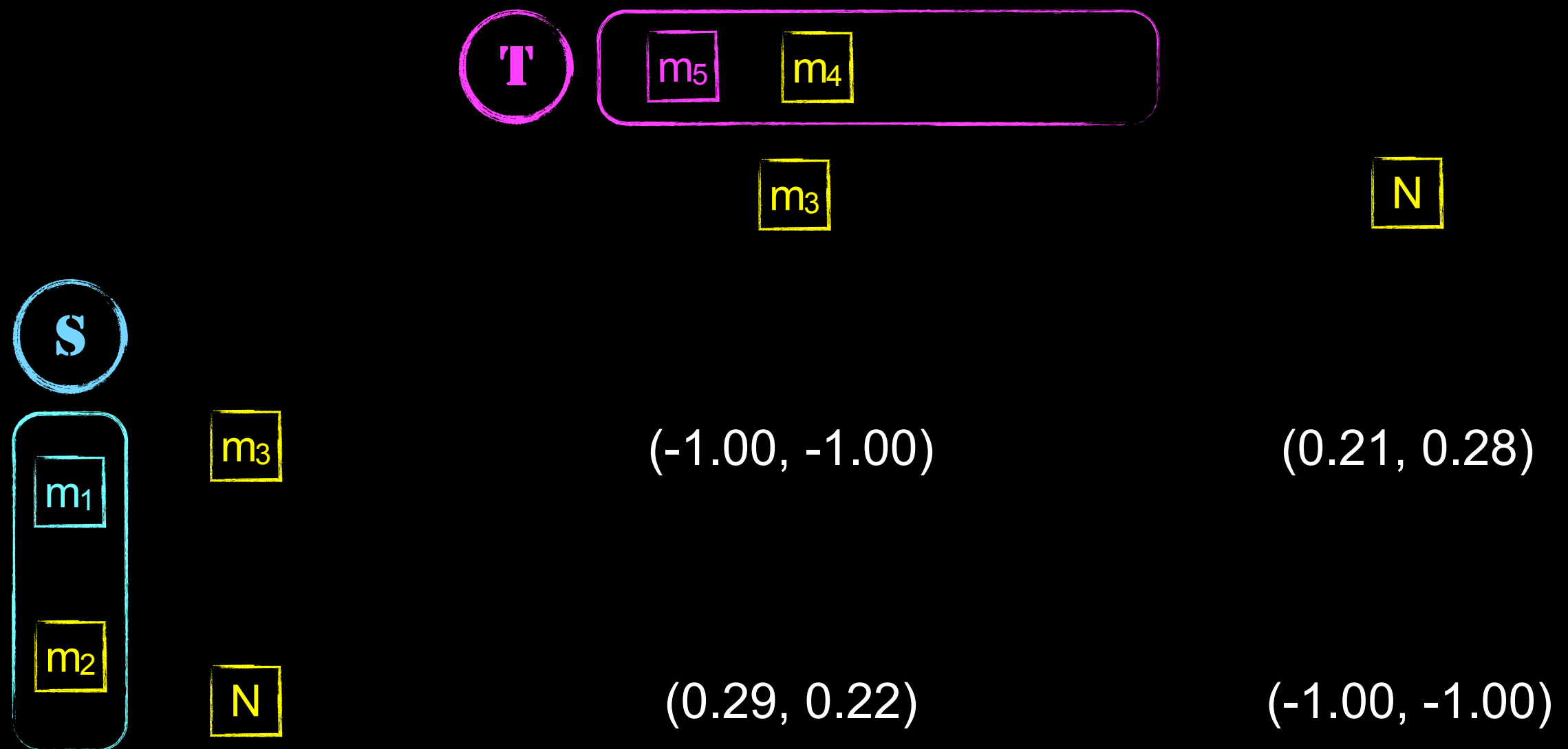
		T					
				m ₅	m ₄		
		m ₂	m ₃				
						N	
S				(-1.00, -1.00)	(0.49, 0.22)	(0.70, 0.30)	(0.70, 0.50)
	m ₃			(-0.49, -0.24)	(-1.00, -1.00)	(0.21, 0.58)	(0.21, 0.28)
	m ₄			(-0.70, -0.80)	(-0.21, -0.58)	(-1.00, -1.00)	(0.00, -0.30)
	m ₂						
	N			(-0.20, 0.00)	(0.29, 0.22)	(0.50, 0.80)	(-1.00, -1.00)

NASH
EQUILIBRIUM

Playing with Refactoring

MODELLING A
NON-COOPERATIVE GAME

..choosing the better strategy



Playing with Refactoring

MODELLING A
NON-COOPERATIVE GAME

..choosing the better strategy

		T	
		m ₅	m ₄
S	m ₃	(-1.00, -1.00)	(0.21, 0.28)
	N	(0.29, 0.22)	(-1.00, -1.00)

Playing with Refactoring

MODELLING A
NON-COOPERATIVE GAME

..choosing the better strategy

		T	
		m ₅	m ₄
S	m ₃	(-1.00, -1.00)	(0.21, 0.28)
	N	(0.20, 0.22)	(-1.00, -1.00)

NASH
EQUILIBRIUM

Playing with Refactoring

MODELLING A
NON-COOPERATIVE GAME

..choosing the better strategy

		T		
		m ₅	m ₄	m ₃
S	m ₃	(-1.00, -1.00)	(0.21, 0.28)	
	N	(0.22, 0.20)	(-1.00, -1.00)	

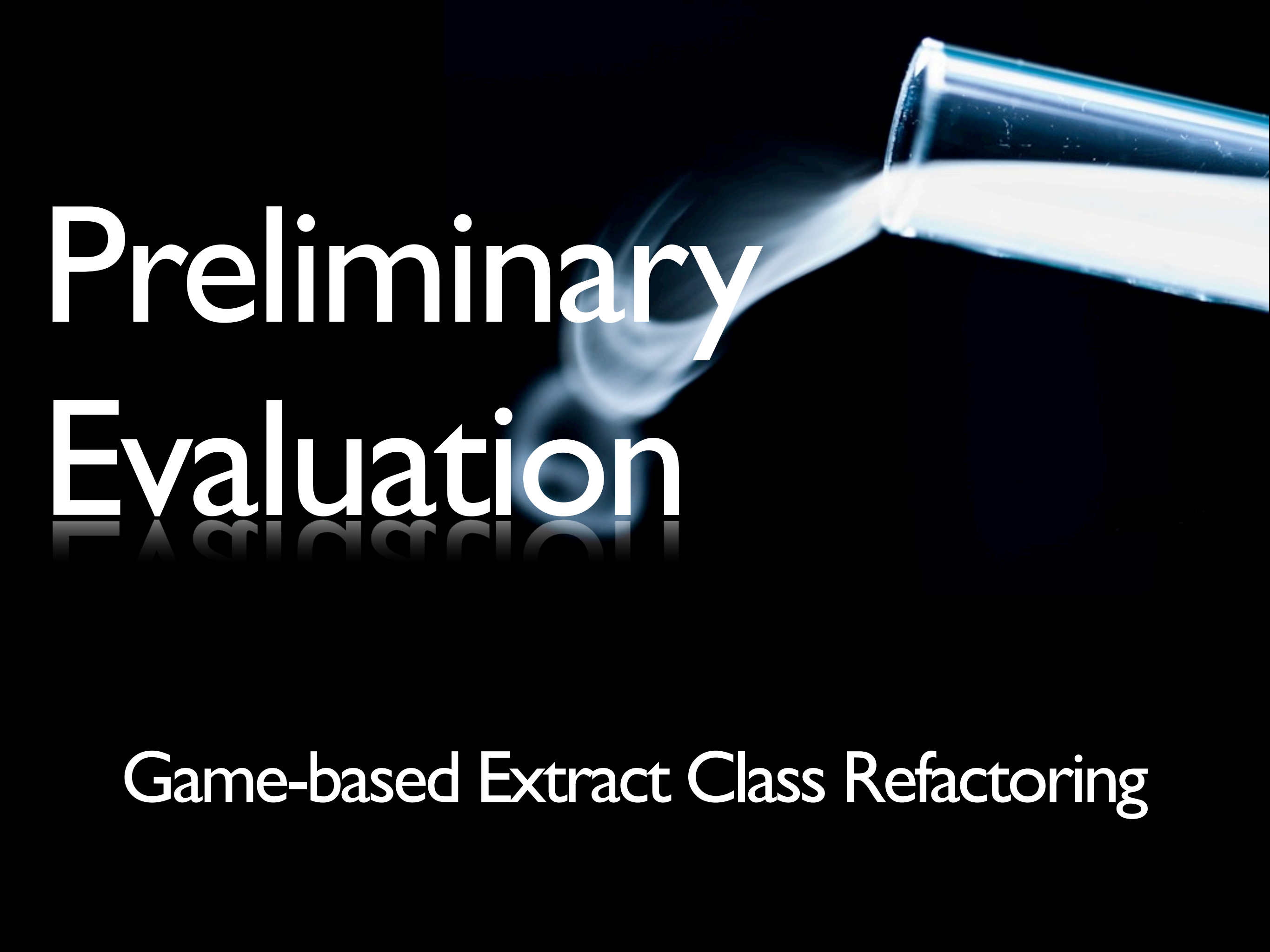
NASH
EQUILIBRIUM

Playing with Refactoring

MODELLING A
NON-COOPERATIVE GAME

..choosing the better strategy





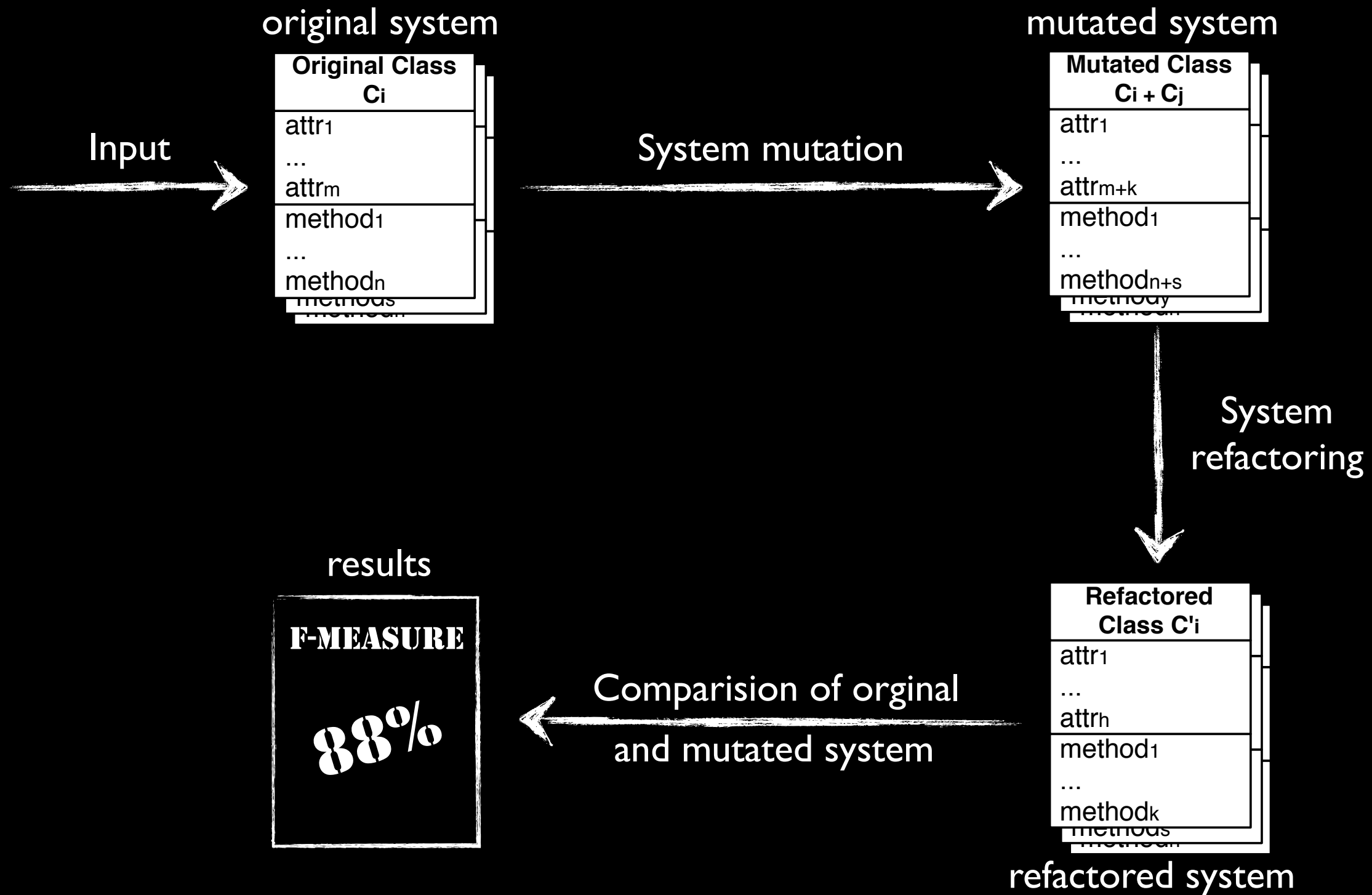
Preliminary Evaluation

Game-based Extract Class Refactoring

Case Study Design

	Goal	Systems	Metrics
RQ ₁	Comparison with Pareto Optimum	ArgoUML, JHotDraw	F-measure
RQ ₂	Comparison with others Extract Class Refactoring approaches	ArgoUML, JHotDraw	F-measure

Experiment Execution



Results

System	Game Theory	Pareto Optimum	MaxFlow MinCut
ArgoUML	90%	88%	77%
JHotDraw	85%	82%	76%



Conclusion and Future Work

Conclusion...

Refactoring... Why?



- Changing software *without* modifying its external behaviour
- Improve non-functional attributes of the software



- Software evolution ... continuous changes
- Changes cause a drift of the original design, reducing its

The first recommendation system that exploits game theory techniques

Game Theory



- is a branch of mathematics widely applied in the social sciences
- capture behavior in strategic situations, in which an individual's success in making choices depends on the choices of others



- a game consists of:

- a set of players (2 or more);
- a set of moves available to those players;
- payoffs for each combination of moves

Game Theory meets SE



Game-based Extract Class Refactoring

Playing with Refactoring

MODELLING A NON-COOPERATIVE GAME

...choosing the better strategy

$0.70 = \text{sim}(m_1, m_2) - \text{sim}(m_1, m_4)$

	m_2	m_3	m_4	N
m_2	(-1.00, -1.00)	(0.49, 0.22)	(0.70, 0.80)	(0.70, 0.50)
m_3	(-0.49, -0.24)	(-1.00, -1.00)	(0.21, 0.58)	(0.21, 0.28)
m_4	(-0.70, -0.80)	(-0.21, -0.58)	(-1.00, -1.00)	(0.00, -0.30)
N	(-0.20, 0.00)	(0.29, 0.22)	(0.50, 0.80)	(-1.00, -1.00)

Conclusion...

Preliminary Evaluation

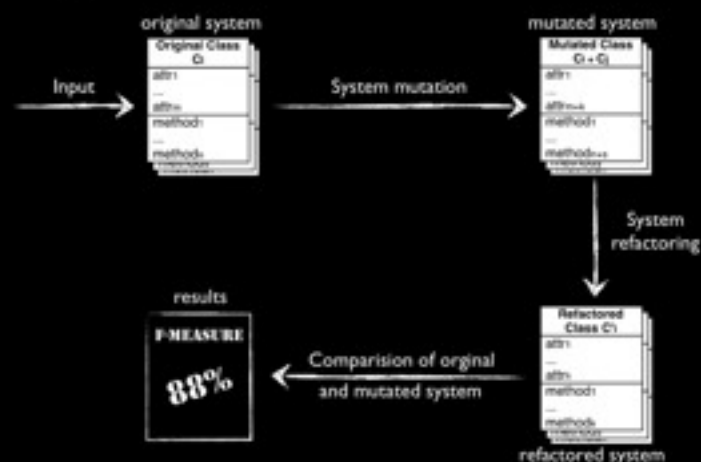
Game-based Extract Class Refactoring

Case Study Design

	Goal	Systems	Metrics
RQ ₁	Comparison with Pareto Optimum	ArgoUML, JHotDraw	F-measure
RQ ₂	Comparison with others Extract Class Refactoring approaches	ArgoUML, JHotDraw	F-measure

Preliminary evaluation of the proposed approach

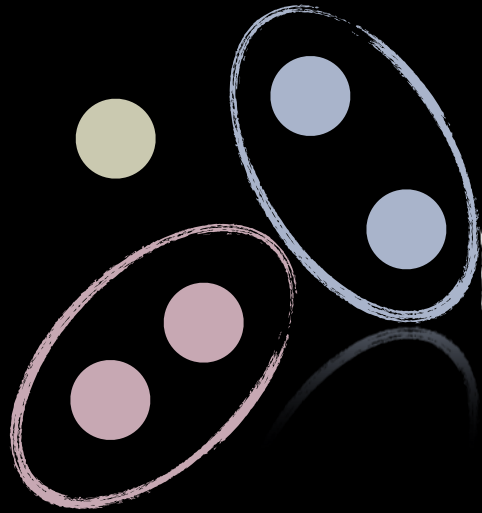
Experiment Execution



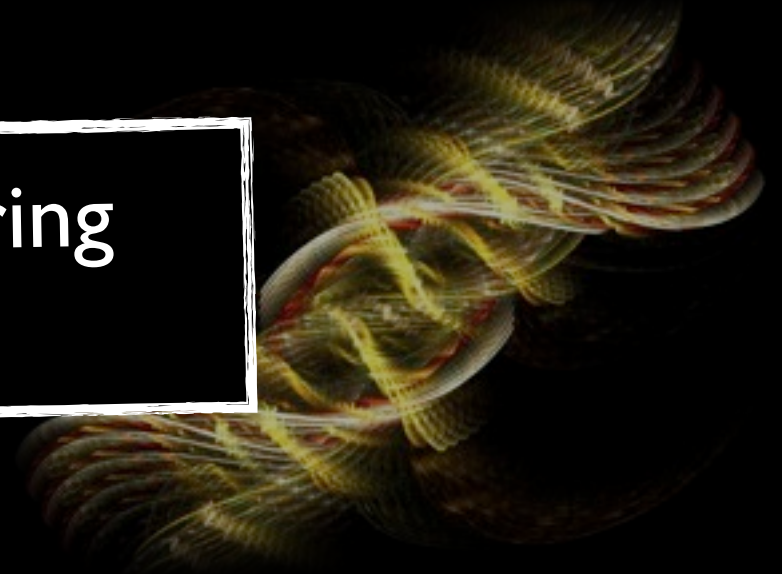
Results

System	Game Theory	Pareto Optimum	MaxFlow MinCut
ArgoUML	90%	88%	77%
JHotDraw	85%	82%	76%

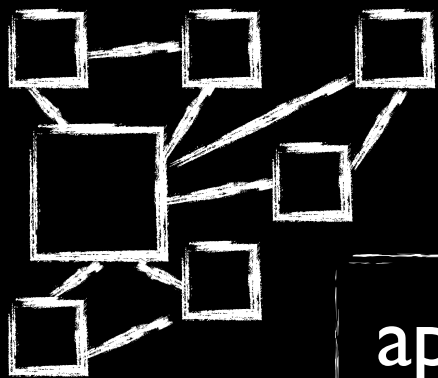
...and Future Work



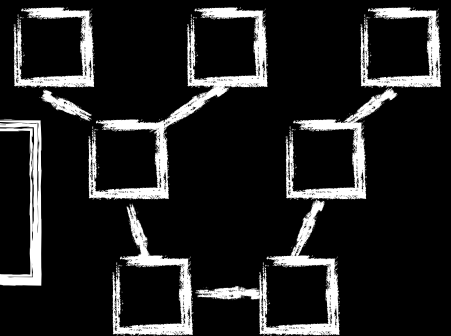
direct comparison with clustering
and search-based approach



investigate about other kind of games, e.g., cooperative game



apply Game Theory to Software Re-modularization



Thank you!

Questions and/or comments

Gabriele Bavota
PhD Student
DMI - University of Salerno
gbavota@unisa.it