Interactive Search Approaches for Requirements Prioritization

Angelo Susi

In collaboration with Alessandro Marchetto, Francis Palma, Giuseppe Scanniello, PaoloTonella

Fondazione Bruno Kessler
Software Engineering Research Unit
Trento, Italy

Outline

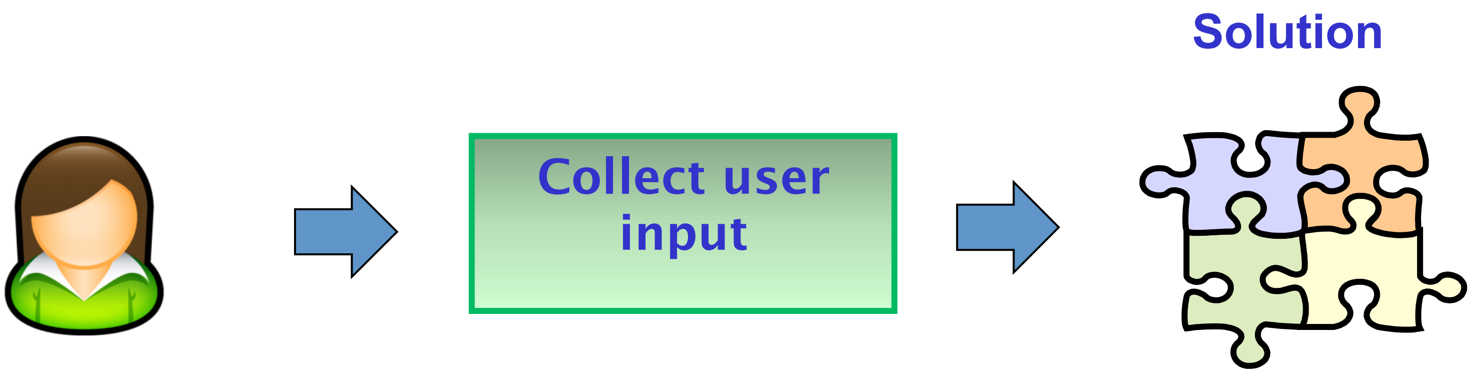
- The problem of requirements prioritization
- The purpose of interaction in prioritization methods
- Exploitation of Interactive Genetic Algorithm
- Applications of the approach to test cases prioritization (ongoing work)

The problem of (requirements) prioritization

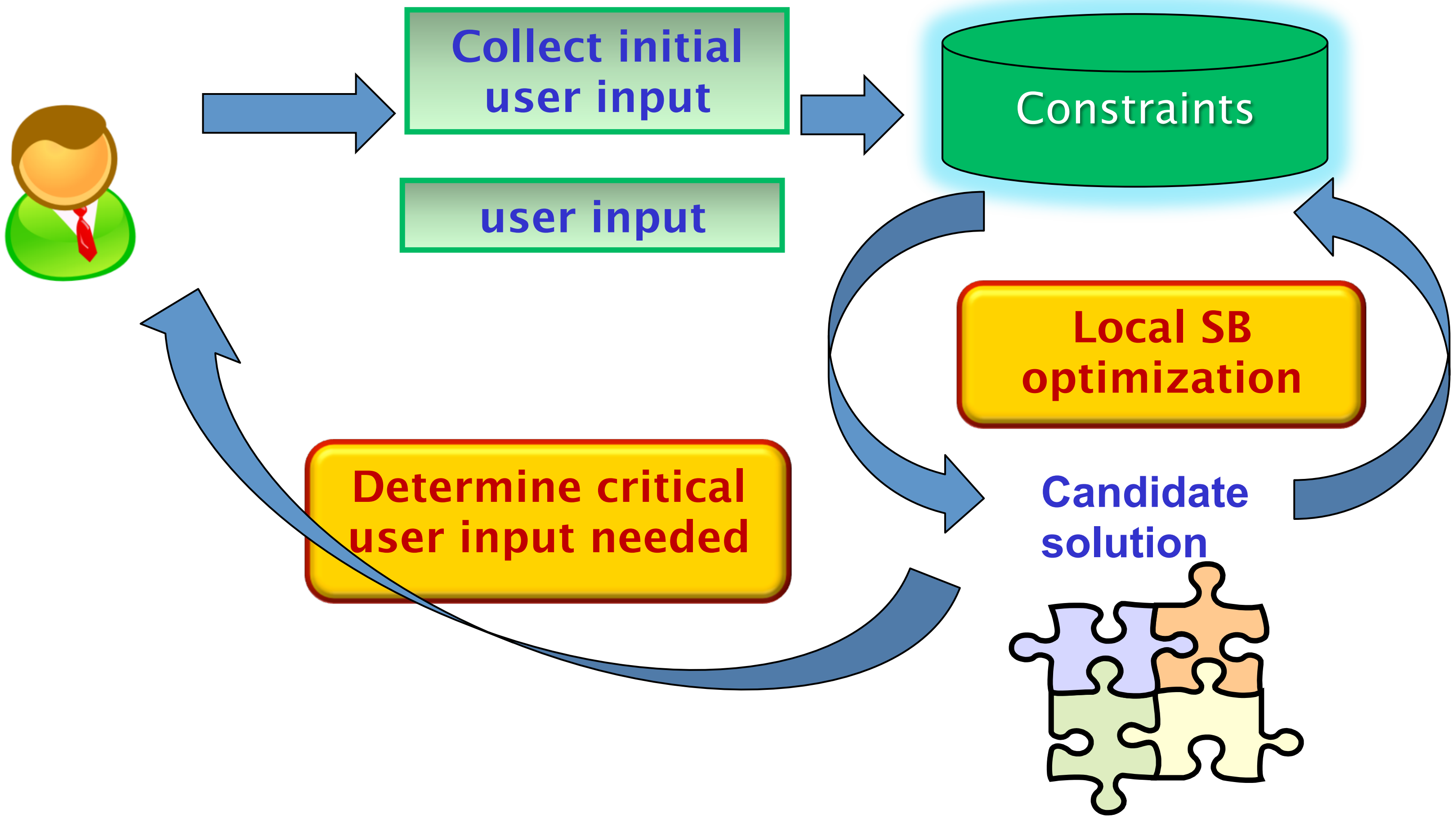
- The activity of finding an order relation on the set of requirements under analysis

- Prioritize according to domain knowledge / constraints concerning
  - available budget
  - time constraints
  - business risks
  - stakeholder expectations
  - technical constraints

A simple Interaction Schema



The Interaction Schema



Why interaction in prioritization

- Increment the effectiveness of the prioritization process via the exploitation of knowledge by the decision makers
  - Decision makers have a lot of "hidden" information that can be "extracted" and exploited
- (Expected) effects:
  - Inclusion of "emerging" knowledge stimulated by the specific subproblem to solve
  - Decrease of the decision making effort
  - Increase of the precision of the result

Some key points

- Important ingredients for interaction are:
  - The process should be able to detect "*critical*" points (such as inconsistencies between constraints and inconsistencies in the feedback)
  - The process should be able to "*express*" the critical point and ask the decision maker(s) to solve it
  - The process should be able to exploit the feedback

Approaches

- Incomplete Analytic Hierarchy Process (IAHP): state-of-the-art pairwise comparison approach, considers only user feedback on the set of alternative requirements, and exploit it to drive the elicitation process

- CBRank: pairwise approach based on Machine Learning techniques that take into account previous user feedback and the domain constraints to drive the process of elicitation of the feedback

- Interactive Genetic Algorithms: use of genetic algorithms to find the solution and drive the elicitation process (that could be based on pairwise comparisons)

(One of) our approach(es)

- Based on *Interactive* Genetic Algorithm (IGA)
  - aims at *minimizing the disagreement* between a *total order of prioritized requirements* and the *various constraints that are either encoded with the requirements* or that are expressed *iteratively* by the user during the prioritization process

The Input

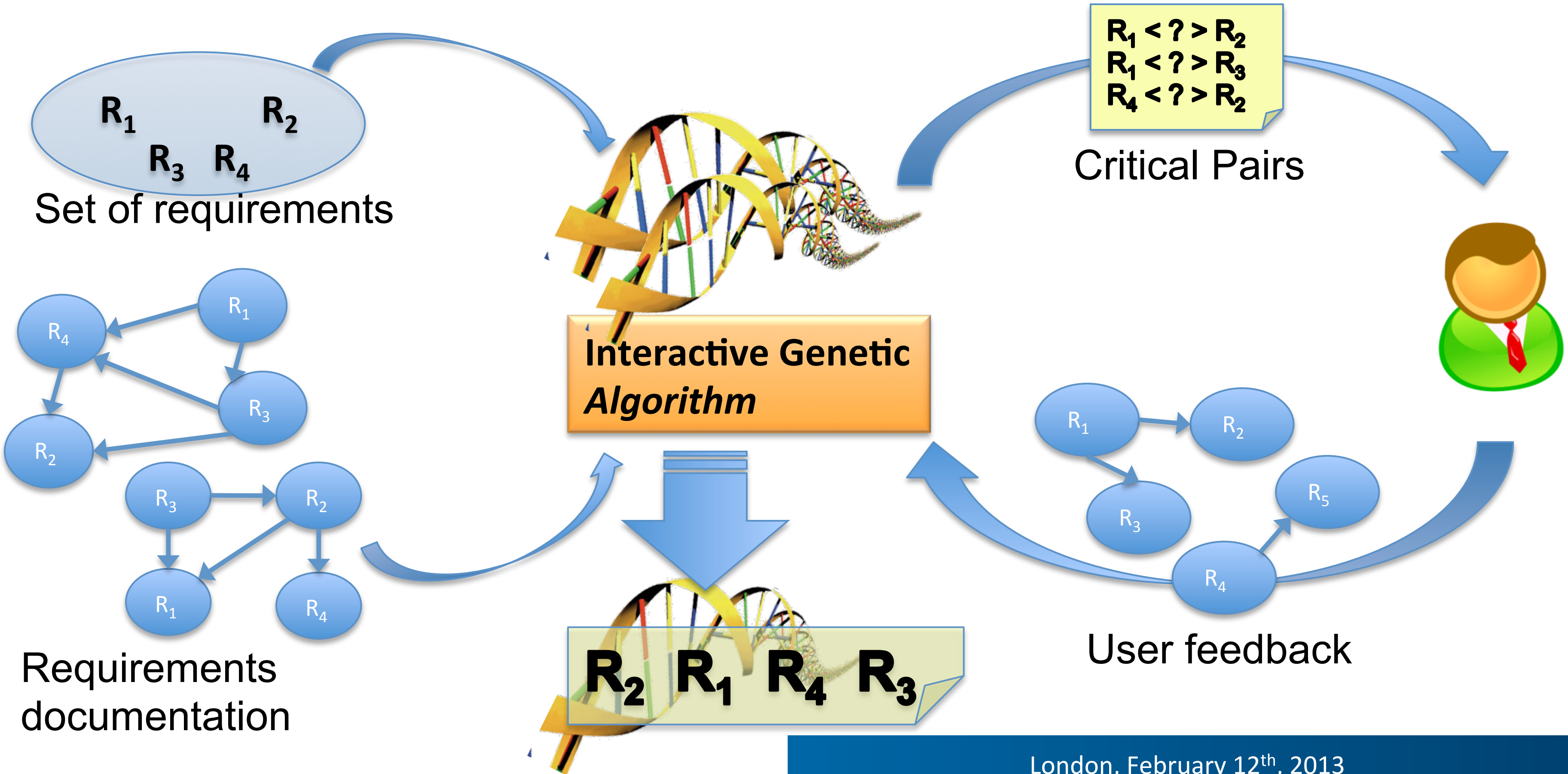
- Set of *Requirements*

- *Requirements documentation* (e.g., cost of the implementation, value for the stakeholders, dependencies between requirements) that can be converted into total or partial rankings of the requirements

- *Evaluation from users* in terms of orderings between pairs of requirements

The process

1. Acquisition and coding of set of Requirements and Documentation
2. Interactive Genetic Algorithm: computation of solutions (individuals), also exploiting evaluations from users
3. Output of the ranking (the most promising individual)



The IGA algorithm: step 2.

2.1. computation of a first set of solutions (individuals)

2.2. identification of **conflicts** (ties) between individuals and constraints

2.3. request of knowledge to users (to decide about conflicts)

2.4. computation of new solutions

- via evolution rules

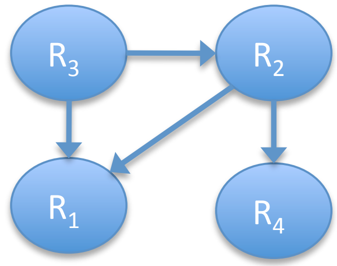
2.5. If max number of iterations

- than exit
- else 2.2

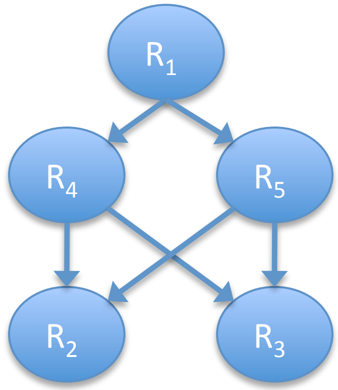
Req. documentation coding into graphs

Transform the **domain knowledge** into graphs

Req	Priorities	Dependencies
R_1	High	R_2, R_3
R_2	Low	R_3
R_3	Low	
R_4	Medium	R_3
R_5	Medium	



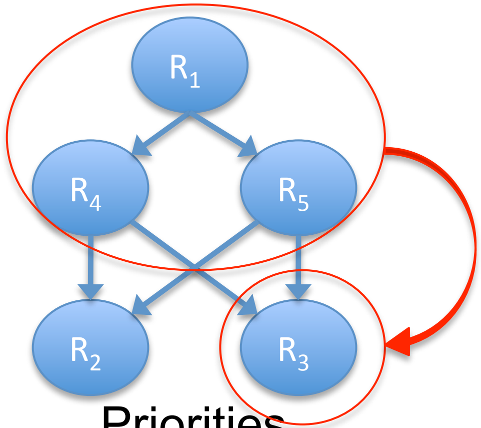
Dependencies



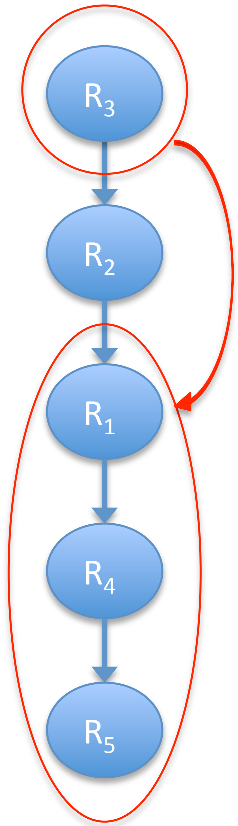
Priorities

Production of individuals and identification of conflicts

Individual ID	Requirements rankings (Individual)	Disagree
Pr_1	$< R_3,R_2,R_1,R_4,R_5 >$	
Pr_2	$< R_3,R_2,R_1,R_5,R_4 >$	
Pr_3	$< R_1,R_3,R_2,R_4,R_5 >$	
Pr_4	$< R_2,R_3,R_1,R_4,R_5 >$	
Pr_5	$< R_2,R_3,R_4,R_5,R_1 >$	
Pr_6	$< R_2,R_3,R_5,R_4,R_1 >$	



Priorities

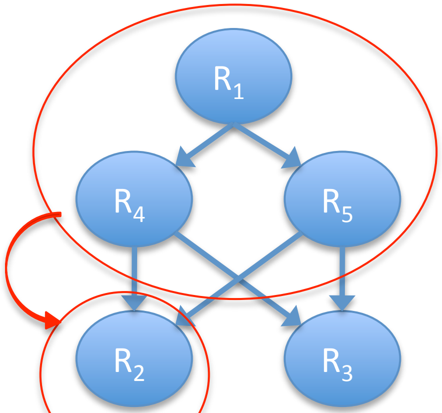


Conflicts = $\{(R_3, R_1), (R_3, R_4), (R_3, R_5)\}$

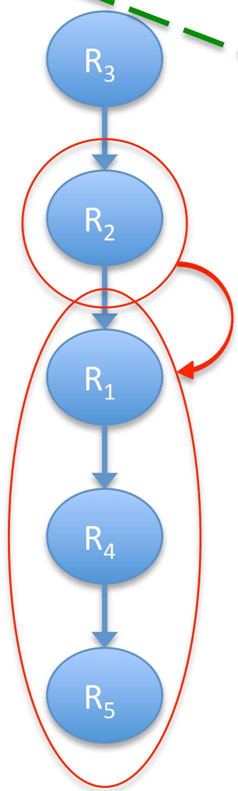
$$dis(pr_1, pr_2) = \{(r,s) \in pr_1^* \mid (r,s) \in pr_2^*\}$$

Production of individuals and identification of conflicts

Individual ID	Requirements rankings (Individual)	Disagree
Pr_1	$< R_3,R_2,R_1,R_4,R_5 >$	6
Pr_2	$< R_3,R_2,R_1,R_5,R_4 >$	6
Pr_3	$< R_1,R_3,R_2,R_4,R_5 >$	6
Pr_4	$< R_2,R_3,R_1,R_4,R_5 >$	7
Pr_5	$< R_2,R_3,R_4,R_5,R_1 >$	9
Pr_6	$< R_2,R_3,R_5,R_4,R_1 >$	9



Priorities



Conflicts = $\{(R_2, R_1), (R_2, R_4), (R_2, R_5), (R_3, R_1), (R_3, R_4), (R_3, R_5)\}$

… and so on …

$$dis(pr_1, pr_2) = \{(r,s) \in pr_1^* \mid (r,s) \in pr_2^*\}$$

Pairs to be evaluated to choose the individuals for feedback

Individual ID	Requirements rankings (Individual)	Disagree
Pr_1	$< R_3,R_2,R_1,R_4,R_5 >$	6
Pr_2	$< R_3,R_2,R_1,R_5,R_4 >$	6
Pr_3	$< R_1,R_3,R_2,R_4,R_5 >$	6
Pr_4	$< R_2,R_3,R_1,R_4,R_5 >$	7
Pr_5	$< R_2,R_3,R_4,R_5,R_1 >$	9
Pr_6	$< R_2,R_3,R_5,R_4,R_1 >$	9

Ranked individuals with respect to disagreement

$PR_1 = < R_3,R_2,R_1,R_4,R_5 >$
vs
$PR_2 = < R_3,R_2,R_1,R_5,R_4 >$

(R_4,R_5)

Candidate pairs to be asked to decision maker

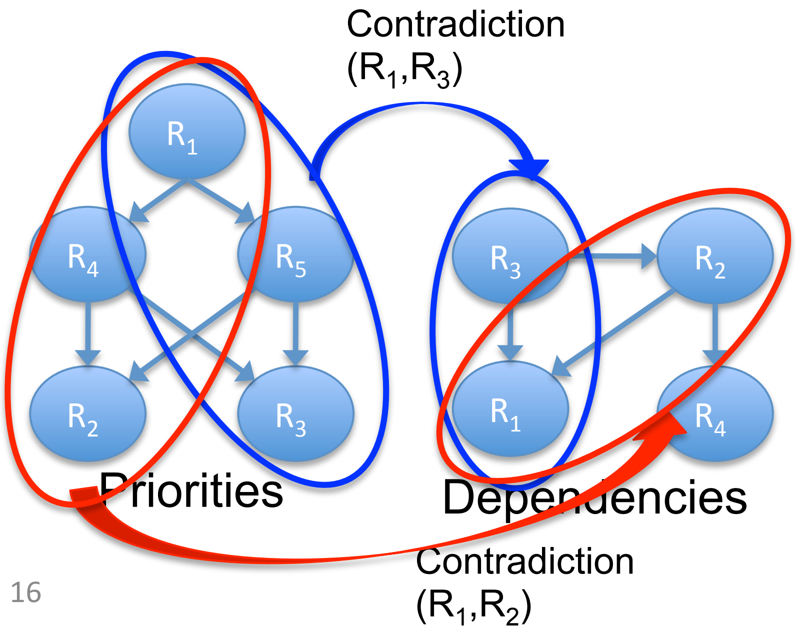
Indiv. ID	PAIRS
Pr_1, Pr_2, Pr_3	(R_4, R_5), (R_1, R_2), (R_1, R_3)
Pr_5,Pr_6	(R_4, R_5)

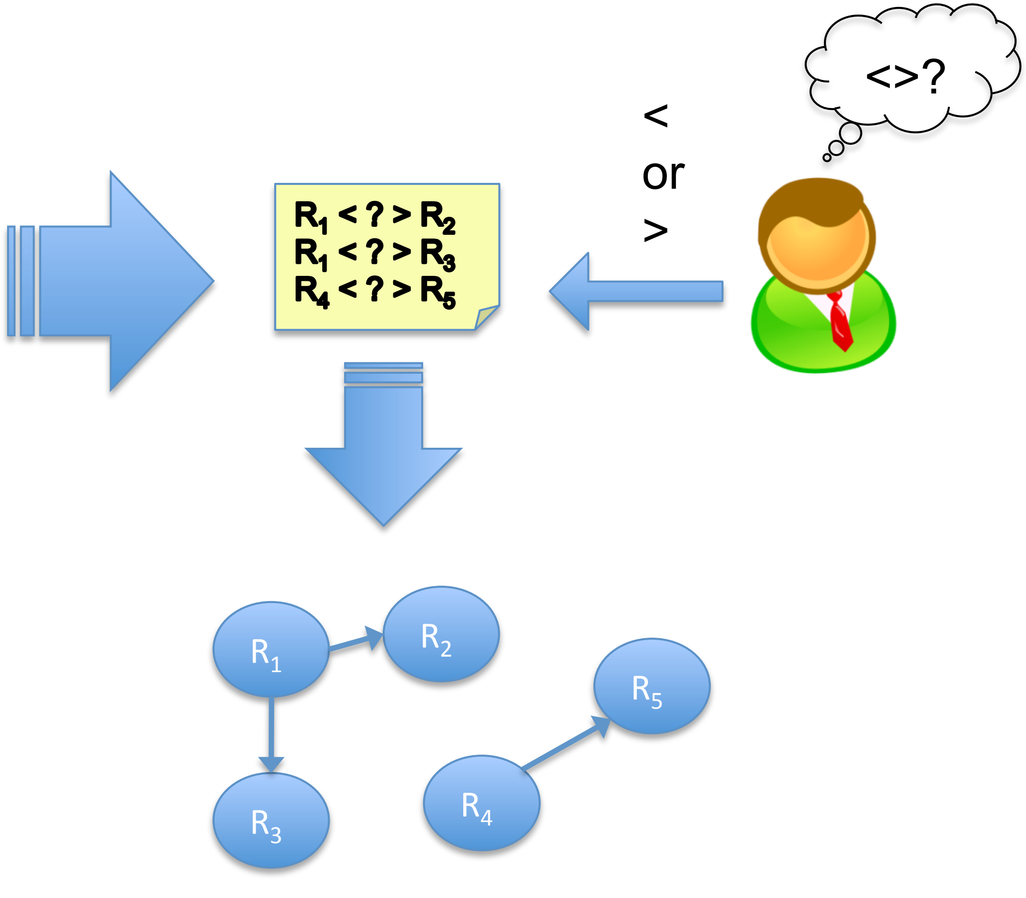
User feedback

TIE	PAIRS
Pr_1, Pr_2, Pr_3	(R_4, R_5), (R_1, R_2), (R_1, R_3)
Pr_5,Pr_6	(R_4, R_5)

Why (R_4,R_5) ?
Nothing is said about (R_4,R_5) in the Priorities and Dependencies graphs

Why (R_1,R_2) and (R_1,R_3) ?

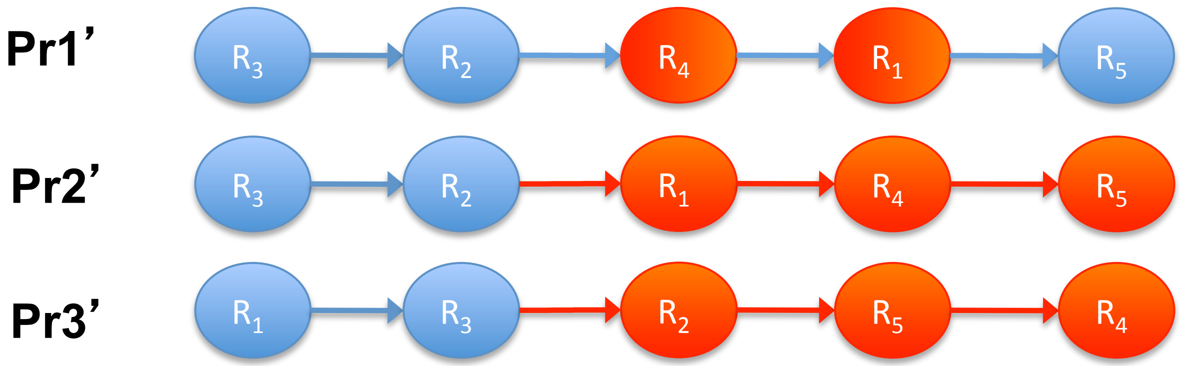




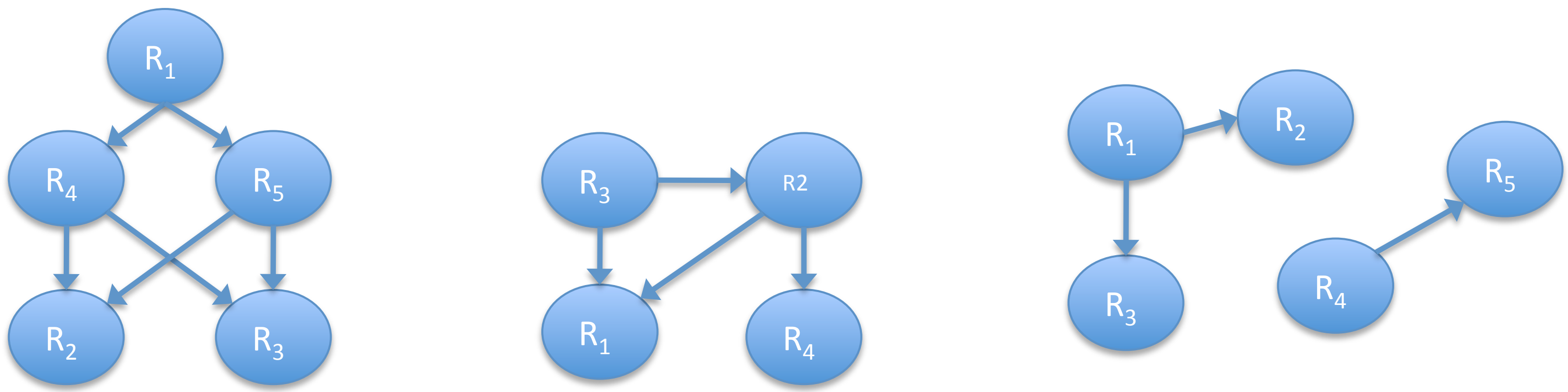
User Preference Graph *eliOrd*

New round of the algorithm

- The new evolved population



- is compared against the new set of constraints graphs



Priorities Dependencies User Preference Graph *eliOrd*

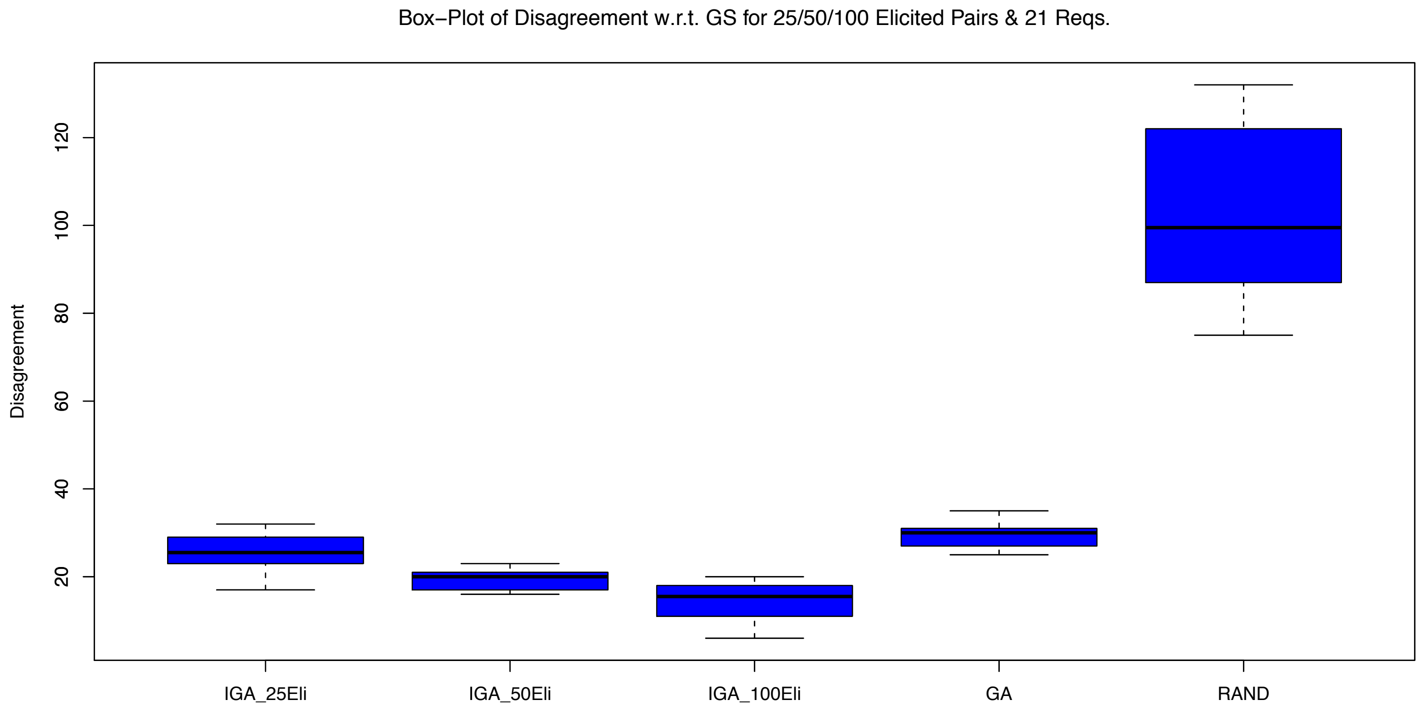
Case Study

- Prioritize requirements for a real software system, as part of the project ACube (Ambient Aware Assistance)
  – designing a highly technological monitoring environment to be deployed in nursing homes to support medical and assistance staff
- After user requirements analysis phase,
  – 60 user requirements and 49 technical requirements
  – Four macro-scenarios have been identified
- A Gold standard from the software architect

Id	Macro-scenario	# of requirements
FALL	Monitoring falls	26
ESC	Monitoring escapes	23
MON	Monitoring dangerous behavior	21
ALL	*The three scenarios*	*49*

Evaluation: Role of interaction

Role of interaction: Does IGA produce improved prioritizations Compared to non-interactive requirement ordering?

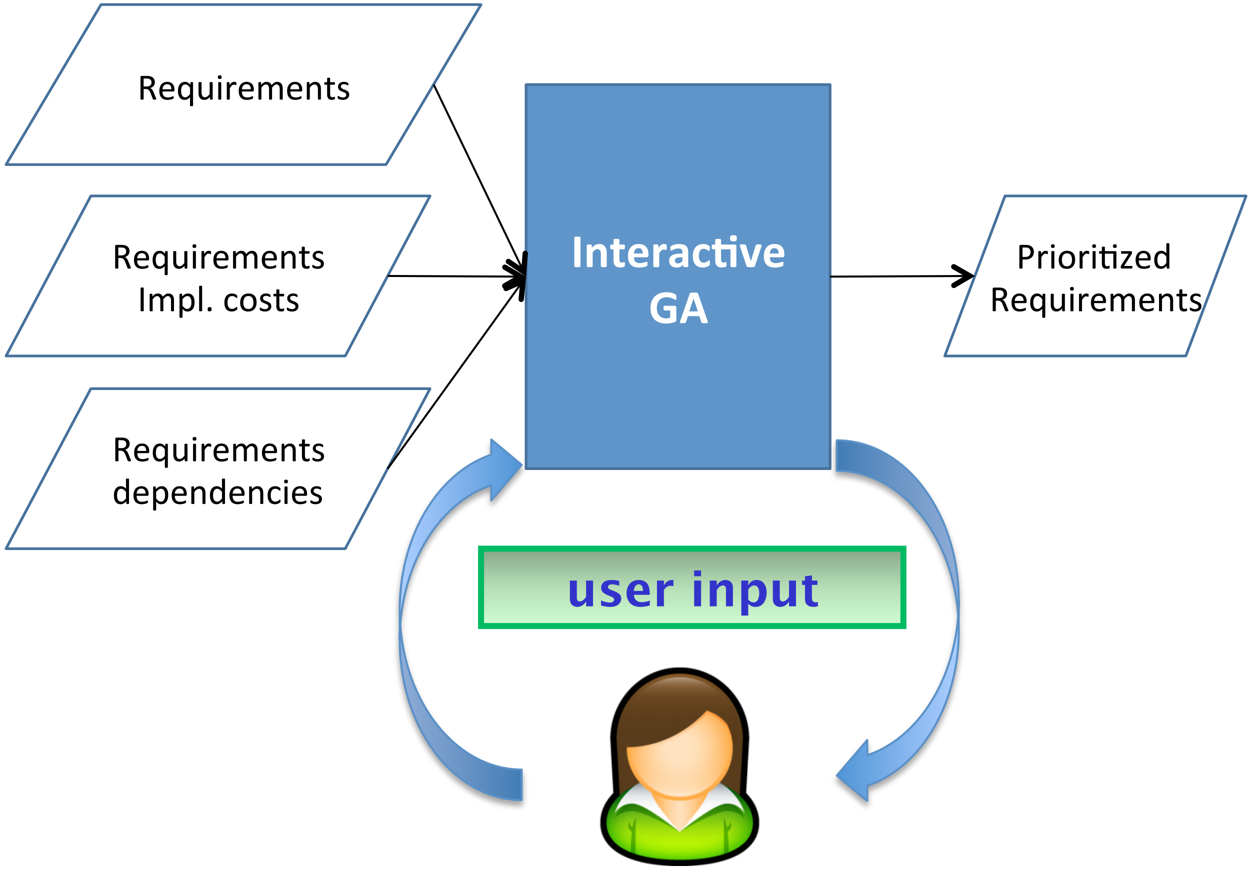


IGA outperforms GA (and RAND), especially when a higher number of pairwise comparisons can be carried out

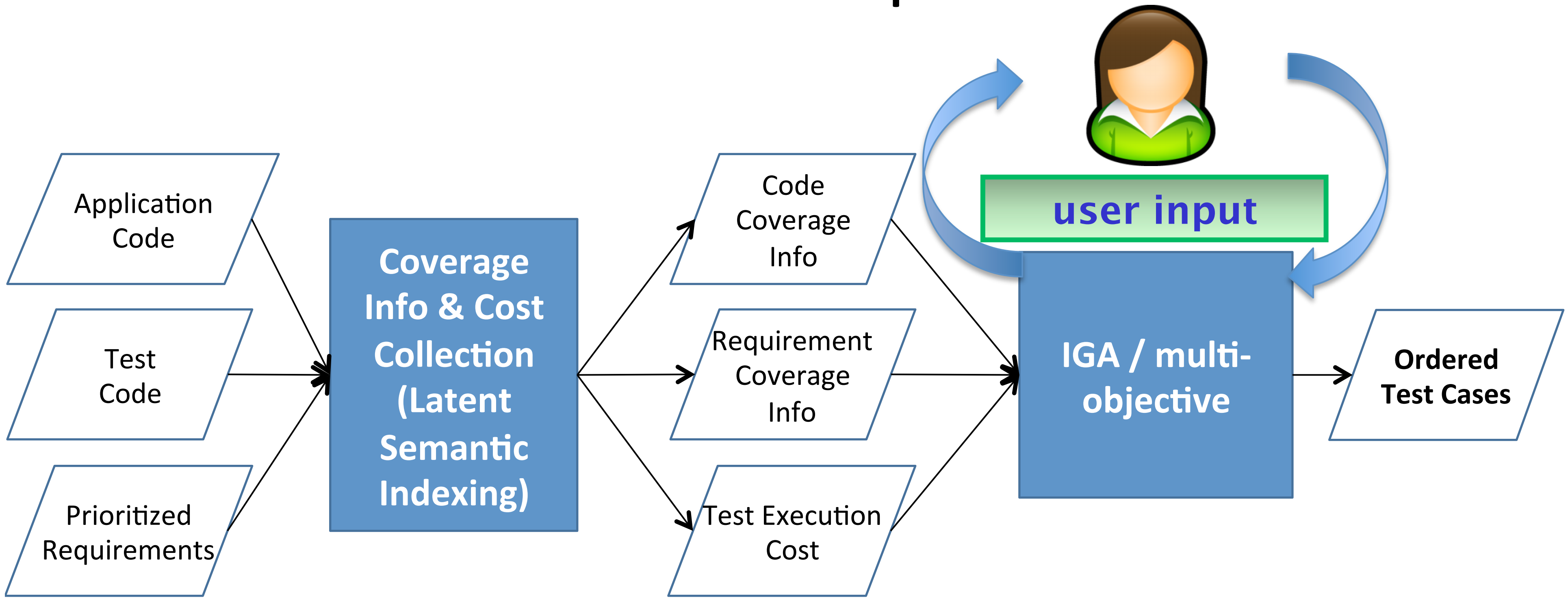
Apply to test case prioritization

- Several objectives to increment the “*value for the user*” (remind the discussion yesterday)
  - Maximize Code Coverage (low level artifact)
  - Maximize “**Most Important Requirements”** Coverage (high level artifact)
  - Minimize Execution Cost
- Advantages
  - Explicitly considers both structural (code) and functional (requirements) dimension at the same time
  - Identifies both technical and business critical faults early
  - Fills gap between low level and high level artifacts by means of traceability

The most important requirements



Test cases prioritization



Collect information about objectives

- Discover traceability links between Test and Source
- Discover Traceability links between Test and Requirements
- Measure execution time of test case

Prioritize

- Test case estimation (Fitness) using Objective Function

Conclusions

- SSBSE is a point of contact between requirements and testing
- Not only, also the exploitation of user knowledge is important in both cases

In the future

TO BE EXPLORED - SSBSE in:

- Risk analysis and mitigation strategies selection
- Normative requirements, were we have to choose among different ways of being compliant with a given law

FONDAZIONE BRUNO KESSLER
RICERCA SCIENTIFICA E TECNOLOGICA

Thank you