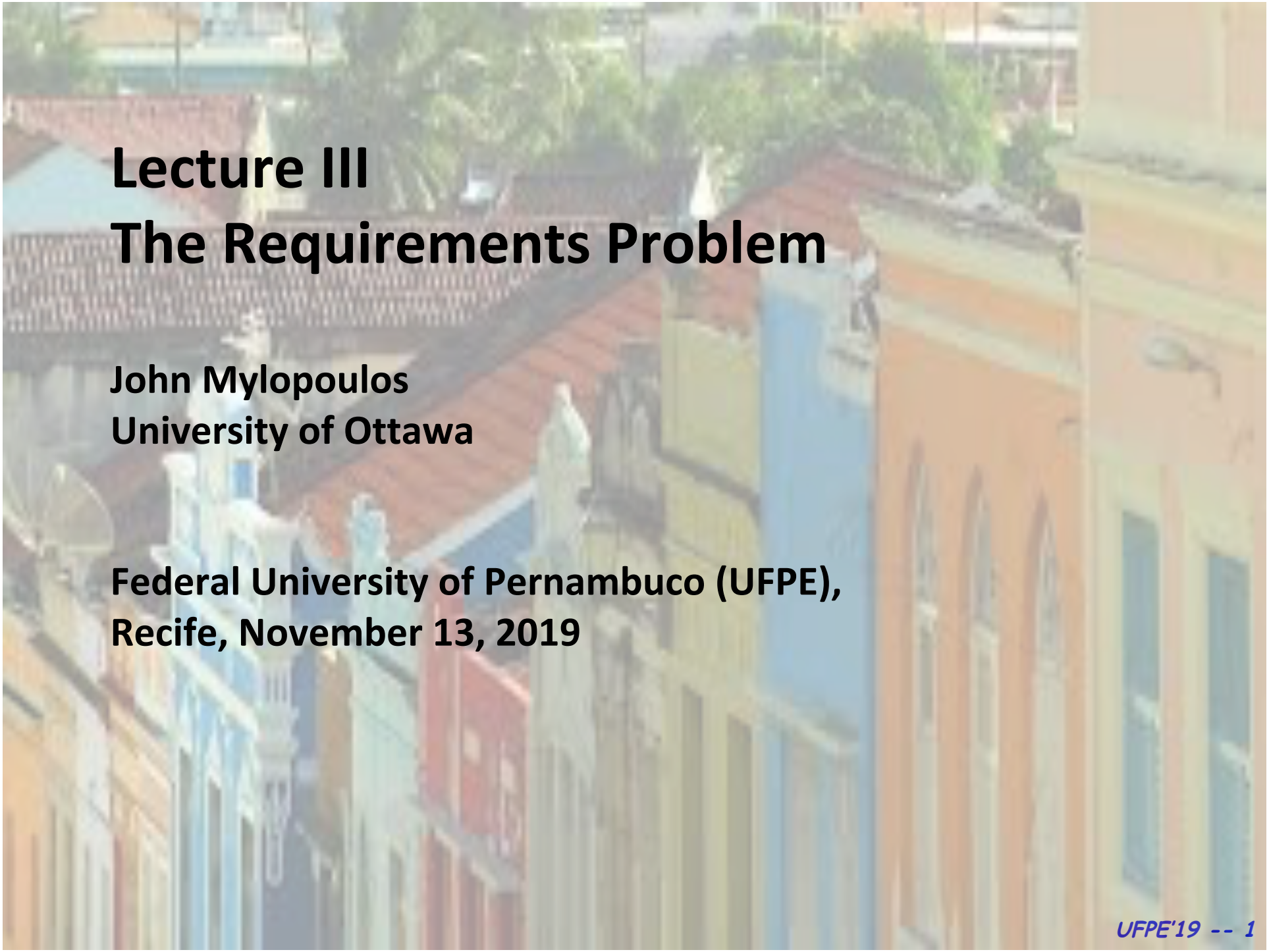


Lecture III
The Requirements Problem

John Mylopoulos
University of Ottawa

Federal University of Pernambuco (UFPE),
Recife, November 13, 2019

The requirements problem (RP)

- The requirements elicited from the stakeholders are statements of need. As indicated earlier, they are invariably informal, conflicting, ambiguous, unattainable, unsound (unnecessary) and incomplete .
- The ***requirements problem*** consists of transforming these needs through a systematic process into a sound, complete, consistent (and possibly formal) specification, consisting of functions, quality constraints, and domain assumptions.
- I'll go over a series of formulations of this problem and discuss the scalability of algorithms for finding solutions, as proposed in the literature.

Tackling the requirements problem

- RP is addressed by building models of problem spaces and then conducting satisfiability/optimization reasoning over such spaces.
- This sounds somewhat exotic for Software Engineering!
- … But it shouldn't be for AI …
- Herb Simon's *Sciences of the Artificial* [Simon69] proposed a framework for a ***Science of Design*** that included all the key ingredients we are using in RE:
  - ✓ ***Goals*** – what is the purpose of the artifact-to-be?
  - ✓ A space of ***alternatives*** for meeting these goals?
  - ✓ ***Search*** among, and ***evaluation*** of alternatives;
  - ✓ …

Jackson and Zave formulation (J&Z)

- In its original formulation [Jackson95], a requirements problem consists of finding a ***specification*** S for a given set of ***requirements*** R and ***indicative environment properties*** E such that

$$E, S \vdash R$$

meaning: "… satisfaction of the requirements can be deduced from satisfaction of the specification, together with the environment properties…" [Jackson95]

- Solution through refinement (as in program refinement): Start with requirements and keep refining them to eliminate mention of non-implementable elements.

Problem refinement

- (Akin to program refinement) Start with requirements and keep refining them to eliminate non-executable elements.
- For instance, (with tm ≡ timeslot)

$\forall$ mtg[Scheduled(mtg)] (Req1)

$\forall$ mtg $\exists$ tm, rm[FdSlot(tm, partL(mtg)) $\wedge$ FdFree(rm, tm)

$\wedge$ Booked(rm, tm, mtg) (Spec)

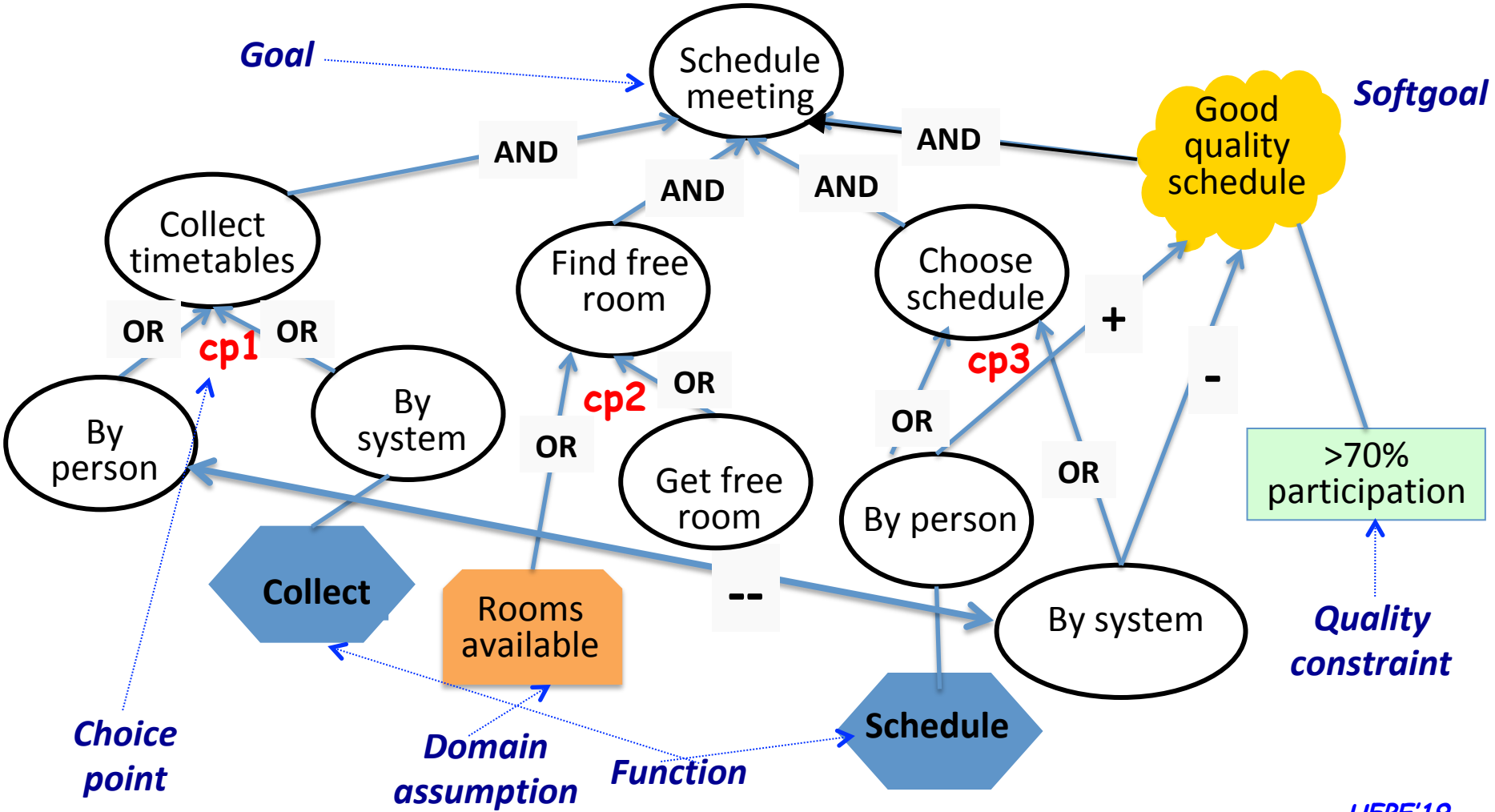
$\forall$ mtg $\exists$ tm, rm[FdSlot(tm, partL(mtg)) $\wedge$ FdFree(rm, tm)

$\wedge$ Booked(rm, tm, mtg) $\Longrightarrow$ Scheduled(mtg)] (DA0)

Spec |= Req1

Requirements as goals (GORE)

- Requirements are now goals and (requirements) problem analysis amounts to incremental goal refinement.



Requirements as goals

- Here, specifications consist of functions, domain assumptions and quality constraints that together satisfy root-level requirements, e.g., for G:ScheduleMtg, one specification is {F:Collect, F;Schedule, DA:RoomsAv, QC: '>70% participation'}
- Unlike J&Z, goal refinement generates ***a space of possible specifications*** and the requirements problem amounts to finding those that satisfy R.
- Finding solutions to GORE problems can be reduced to SAT solving [Sebastiani04], scales for goal models of size ***O***(1K).
- A much more expressive language (KAOS) was proposed for GORE in [Dardenne93], subsuming FOL and LTL.

Formalizing goal models [Sebastinani04]

- To reason with goal models, we first need to formalize the possible truth values of goals and the meaning of relationships:
- Goals can have one or more of the following values S(atisfied), D(enied), PS, PD (for partially satisfied/denied).
- Goal relationships are formalized in terms of axiom patterns, e.g.,

$AND(G, \{G_1,\cdots, G_n\}) \wedge S(G_1) \wedge \cdots \wedge S(G_n) \Longrightarrow S(G)$

$OR(G, \{G_1,\cdots, G_n\}) \wedge (S(G_1) \vee \cdots \vee S(G_n)) \Longrightarrow S(G)$

$AND(G, \{G_1,\cdots, G_n\}) \wedge (D(G_1) \vee \cdots \vee S(G_n)) \Longrightarrow D(G)$

…

$+(G, G') \wedge S(G) \Longrightarrow PS(G')$

$-(G, G') \wedge S(G) \Longrightarrow PD(G')$

…

Reasoning with goal models

- Given a goal model, we can generate axioms using the axiom patterns shown earlier.
- For example, for the ScheduleMtg (SM) goal model we generate axioms such as

$$\text{AND}(SM, \{CT, FFR, CS, GQS\} \wedge S(CT) \wedge S(FFR) \wedge S(CS) \wedge S(GQS) \Longrightarrow S(SM)$$

$$+(BP, GQS) \wedge S(BP) \Longrightarrow PS(GQS)$$

- **Reasoning bottom up**. Given truth values for some leaf goals, are root goals satisfied? Use axioms to propagate truth values towards goal model roots.
- **Reasoning top down**. Is there a set of assignments to leaf goals that makes root goals S(atisfied)? Use a SAT solver to answer this question, with input the axioms generated for a particular goal model.

Off-the-shelf reasoners

- When you come up with a new type of model, e.g., goal model, entity-relationship model, business process model, you need to propose a reasoning (aka analysis) technique otherwise your models won't be useful for real world applications.
- To do so, you need to formalize the elements of your models, see earlier slides, but also to map the reasoning you want to do to a problem an off-the-shelf reasoner can solve.
- Off-the-shelf reasoners have been developed over decades of research, they do a lot better than anything an RE researcher can design and implement.

Solvers and checkers

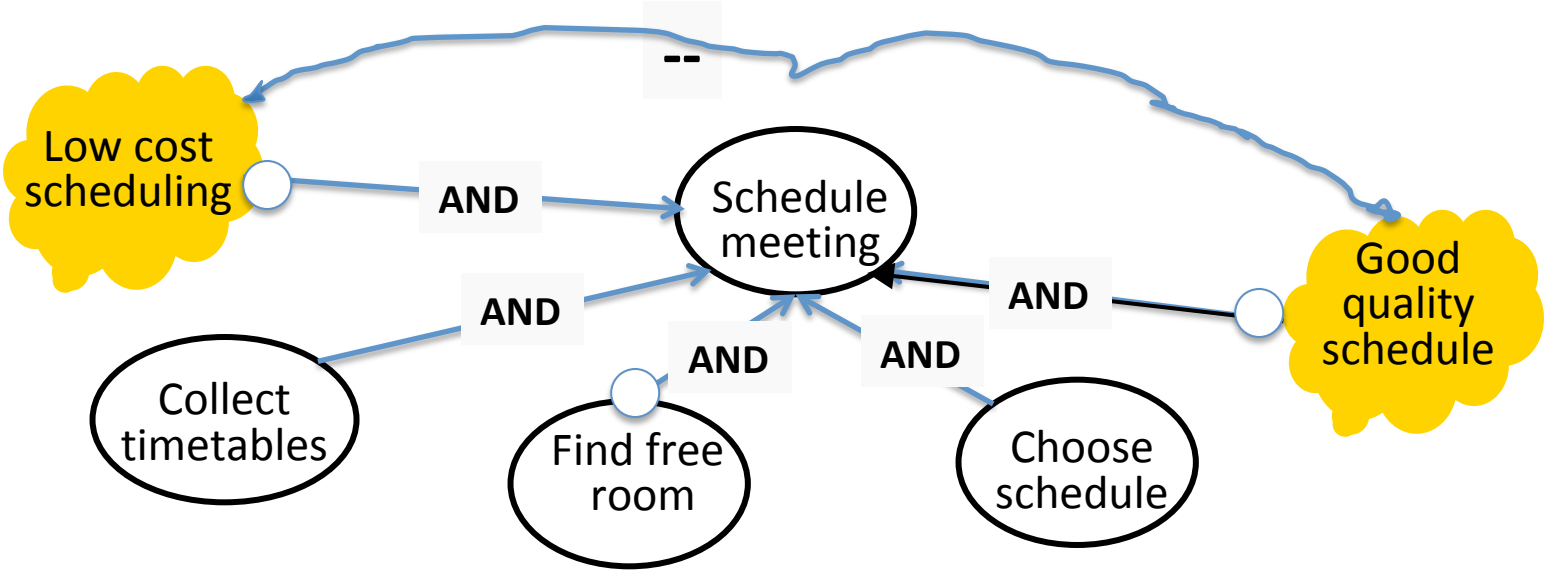
- ***Solvers*** solve satisfiability (SAT) problems for near-propositional logics such as:
  - ✓ SAT solvers solve SAT problems for propositional logic;
  - ✓ SMT (Satisfiability Modulo Theories) solvers solve SAT problems for a decidable FOL theory (e.g., arithmetic over rational numbers)
  - ✓ OMT (Optimization Modulo Theories) solvers solve optimization problems over a decidable FOL theory.
- ***Model checkers*** find counter-examples to a theory. For example, you have a model of insurance claim processing and you claim that no insurance claim can be claimed twice. You formalize your model and the claim as a theory and you feed it to a model checker who finds a counter-example where a claim is claimed twice with two different insurance companies.

Expressiveness vs scalability

- Research on Description Logics (such as OWL) in Knowledge Representation (AI) developed a rich theory on Expressiveness vs Tractability for description logics [Brachman84], [Borgida96].
- In RE, ***tractability*** is not an issue, all the variants of the RP considered in this presentation are intractable; rather, the ***scalability*** of solvers is important: solvers need to "scale" to real world-size requirements problems.
- For the requirements problem, this means problems of size ***O***(1K).

Preferences and priorities (P&P)

- Preferences are "nice-to-have" requirements. Among them, there can be binary priority relationships.



- Low cost >> Find free room (priorities)
- Low cost >> Good quality schedule

P&P problems

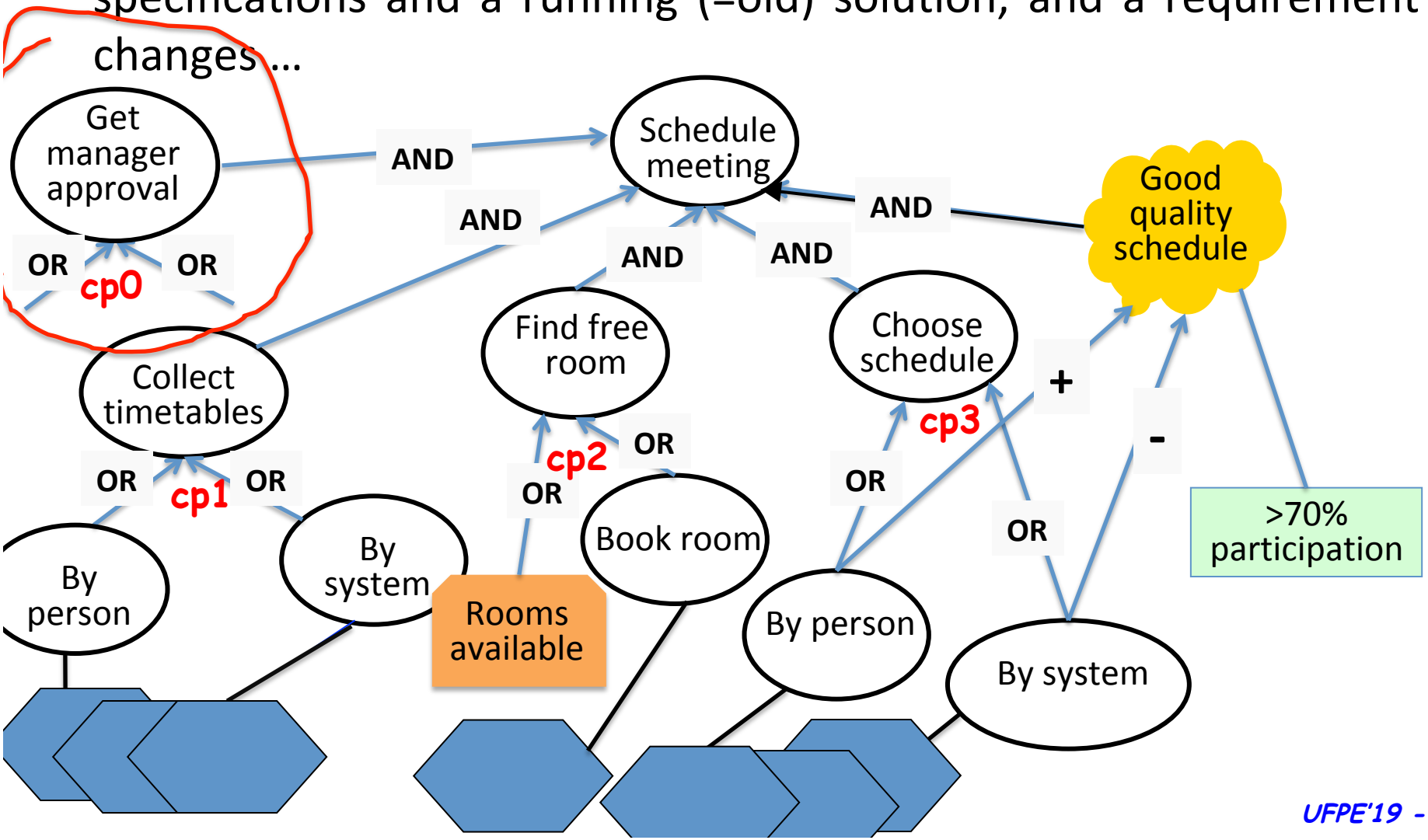
- Now a solution consists of a specification that satisfies all mandatory goals and a ***maximal consistent subset of preferred ones***, with no better solution wrt priorities.
- The requirements problem is now an optimization problem, rather than merely a satisfaction one.
- ***Note***: The semantics of preferred requirements is adopted from AI planning and is different from that of optional requirements. For example, if I have a problem consisting of two consistent requirements {R1, R2} each admitting a single solution, there is one solution if R1, R2 are preferences (namely, {R1, R2}, but 4 solutions if they are optional (namely {}, {R1}, {R2}, {R1, R2}) .

Solving P&P problems

- One way to tackle this version of the requirements problem is to adopt AI planners (another class of solvers!). However, several features of planners are best used during design, rather than RE [Liaskos10].
- Another way is to develop search algorithms from first principles [Jureta10].
- In either case, intractability is a given, while scalability is an open question.
- [Ernst10] uses local search techniques with good performance results that improve over naïve search.
- Local search techniques constitute another class of off-the-shelf heuristic solvers that find satisficing solutions.

Evolving requirements problem (ERP)

- Suppose now we have an architecture that implements several specifications and a running (=old) solution, and a requirement changes …

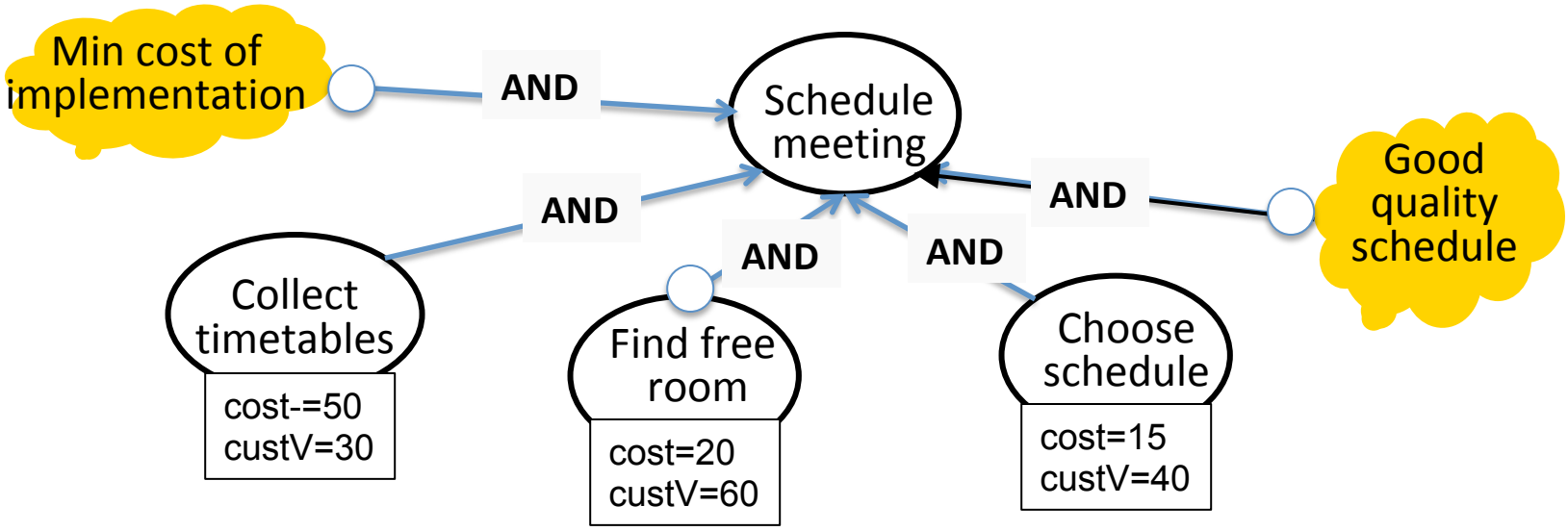


Solving the IRP

- All we need to do is run our GORE/P&P/… search algorithm for solving the ***new*** requirements problem, right? …
- Not quite, if we want to:
  - ✓ Maximize familiarity – use as much as possible the old solution (user perspective)
  - ✓ Minimize effort – minimize the number of new functions that need to be implemented (vendor perspective)
- We need algorithms here that search for repairs when the requirements problem "breaks" due to new requirements.
- [Ernst11] studies a class of such algorithms using Truth Maintenance Systems (ATMS) with not-so-positive results.

RP cum optimization (RPO)

- Suppose now we further extend our goal language to allow associated cost, customer value and other attributes to each goal, and also allow optimization (min/max) for goals



Solving RPO

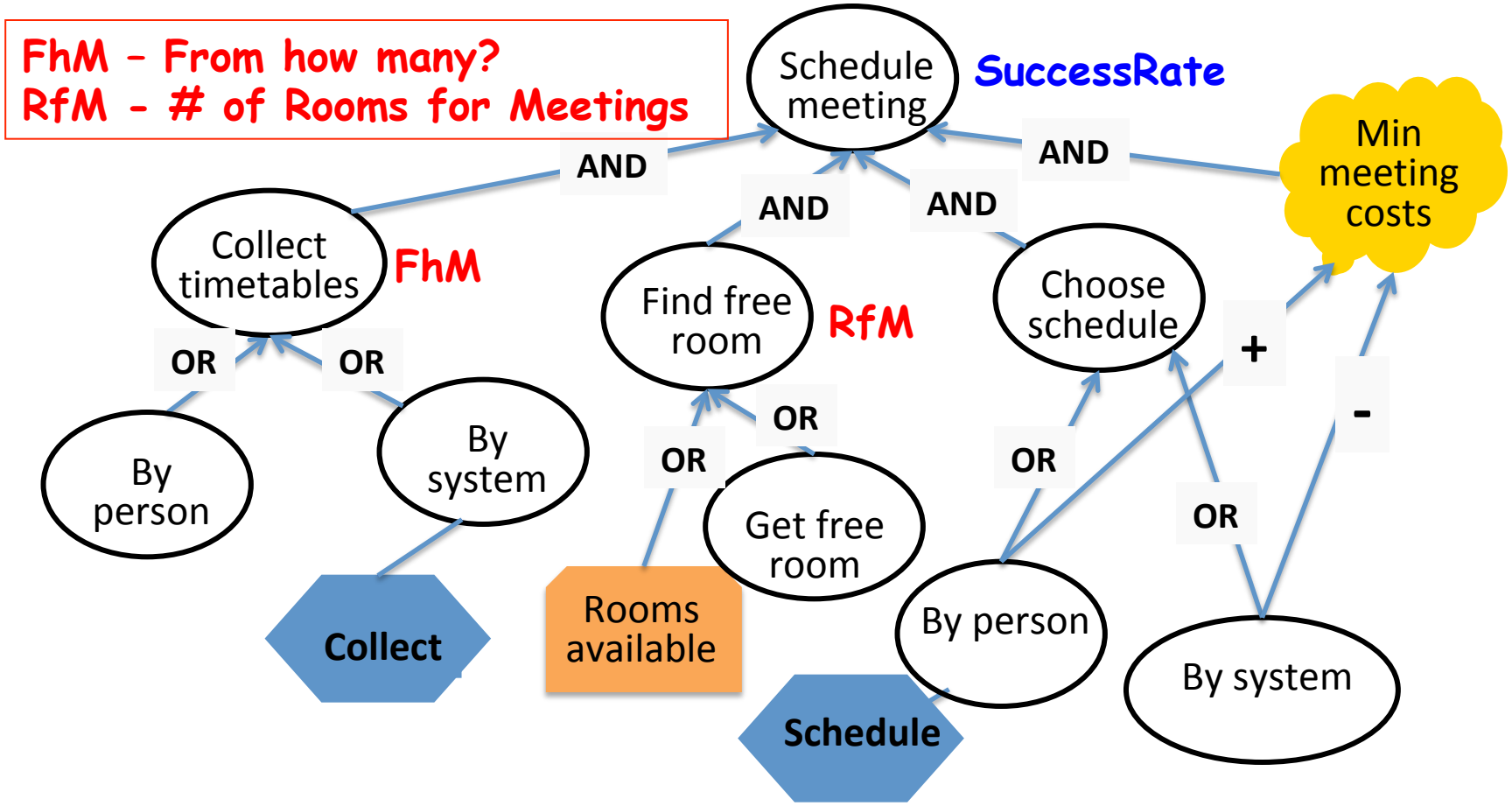
- We need now solvers with richer native languages than SAT solvers.
- Chi Mai Nguyen [Nguyen15] used Optimization Modulo Theories/Satisfiability Modulo Theories (aka OMT/SMT) solvers for solving such problems.
- In particular, she used OptiMathSAT [Sebastiani15] which supports linear arithmetic over rationals, as well as optimization over multiple objective functions, either singularly or lexicographically.
- Our experiments suggest that OMT/SMT solvers scale well, but are very sensitive to the objective functions over which they optimize.

The next release problem (NPR)

- This is a variant of the evolving requirements problem: Given an existing system and a set of new mandatory and/or preferred requirements, you want to find a specification for the next release of the system that optimizes several objective functions, such as value-to-customer, cost-to-implement, etc.
- Fatma Başak Aydemir [Aydemir18] formulated NPR in terms of extended goal models, using the OptiMathSAT OMT/SMT solver as search backend.

The adaptation problem

A specification fulfill its requirements depending on ***control variables*** and ***indicators*** that determine respectively resource allocation, quality of output, etc.



The adaptation problem

- To define the adaptation problem, we also need to define how indicators depend on control variables,

  e.g., $F(FhM,RfM,SuccessRate,CostPerMtg) = 0$

  $G(FhM,RfM,SuccessRate,CostPerMtg) \geq 0$

- Now suppose that the system is monitoring its performance and some requirements are failing (or, equivalently, some indicators are out of bounds).
- We need to find a new set of values for control variables that "restores" failed requirements in an optimal way.
- Again, this is a search problem that can be solved with a OMT/SMT solver.
- Kostas Angelopoulos has addressed this problem, using an OptiMathSAT-based tool [Angelopoulos16].

A more careful look at refinement

- All the variants of the requirements problem presented here (… or elsewhere) use the notion of ***refinement*** to derive incrementally a specification from stakeholder requirements.
- According to the dictionary, 'refinement' means 'the process of removing impurities, defects or unwanted elements from something'.
- In the case of goal models, the defect removed is that a requirement is not operationalizable (aka non-atomic).
- But there are other kinds of defects, just look at the IEEE standard [IEEE98], including: ambiguity, conflict, unattainability, incompleteness, too strong/too weak, unnecessary, …

A refinement calculus for requirements

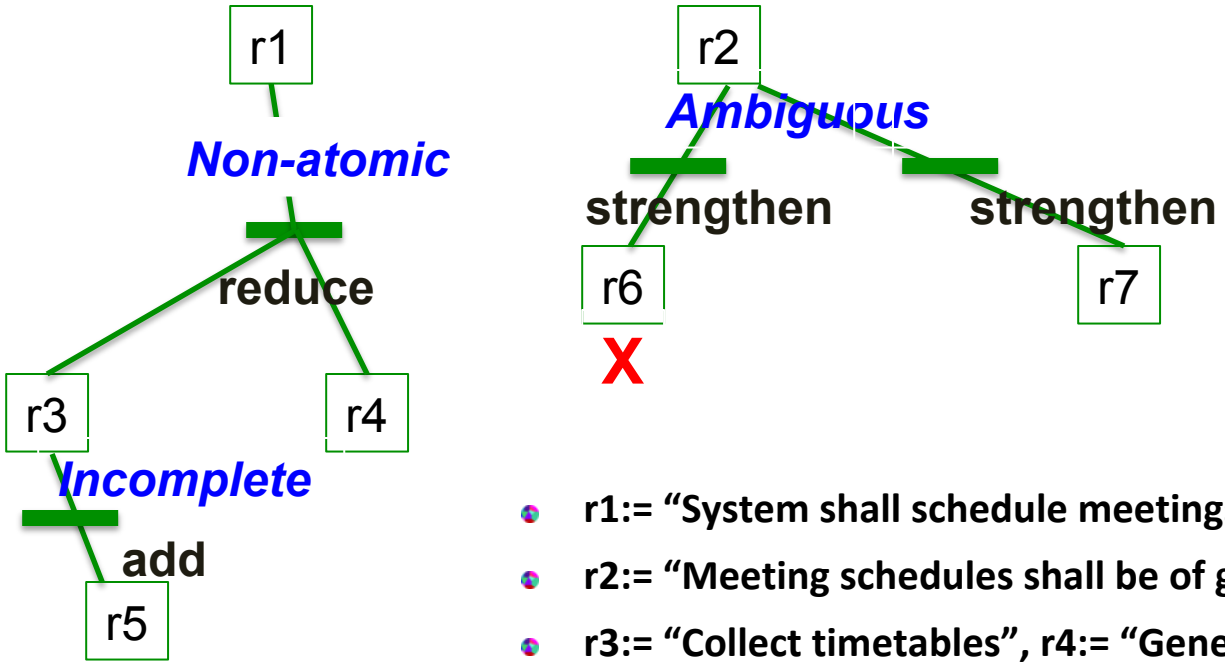
- Now we introduce a larger set of refinements, including those in goal analysis:
  - ✓ Strengthen(r): refines r into r' such that r' ⇒ r.
  - ✓ Weaken(r): refines r into r' such that r ⇒ r'.
  - ✓ Reduce(r): refines r into r1, ⋯, rn such that r1 ∧ ⋯ ∧ rn ⇒ r
  - ✓ Add(r): refines r into r' such that requirement r has not been dealt with until r' has.
  - ✓ Resolve({r1,⋯,rn}): refines r1,⋯,rn into r1',⋯,rm' such that each ri' i=1,⋯,m there is some rj, j=1⋯n such that rj ⇒ ri'.
- This is work-in-progress [ElRakaiby19]

Defect types

- Attacks on a requirement point to defects, and justify refinements.
- Our proposal supports several types, inspired by the IEEE standards on requirements specifications [IEEE93]:
  - ✓ Non-atomic(r): there is no function that operationalizes r.
  - ✓ Ambiguous(r): r has multiple interpretations.
  - ✓ Unattainable(r): r can't be fulfilled.
  - ✓ Unjustified(r): unclear why is r needed.
  - ✓ TooStrong/TooWeak(r)
  - ✓ Rejected(r)
  - ✓ Conflicting({r1,⋯rn})
  - ✓ Incomplete({r1,⋯rn})
- For each attack type, there is at least one operator that can be applied to eliminate the defect pointed out by the attack.

Refinement graphs

A refinement graph consists of nodes (requirements) and hyper-edges (refinements)



- **r1:= "System shall schedule meetings upon request"**
- **r2:= "Meeting schedules shall be of good quality"**
- **r3:= "Collect timetables", r4:= "Generate a schedule"**
- **r5:= "timetable data are confidential"**
- **r6:="≥70% participation rate"**
- **r7:="≥80% participation rate"**

Towards a theory of RE

- The RP manifests itself in many forms and variants thereof.
- We envision a theory of RE that consists of alternative formulations of the RP adopting languages with different levels of expressiveness and includes results on the scalability of tools that find solutions.
- Such a theory is to be founded on satisfiability/ argumentation semantics, and grounded on SAT/SMT/OMT/ ARGT solvers that serve as back-ends to the search for solutions.

References

- [Angelopoulos16] Angelopoulos K., Aydemir F., Giorgini P., Mylopoulos J., "Solving the Next Adaptation Problem with Prometheus", 10th IEEE International Conference on Research Challenges in Information Science (RCIS'16), Grenoble, June 2016.
- [Aydemir18] Aydemir F., Dalpiaz F., Brinkkemper S, Giorgini P., Mylopoulos J., "Solving the Next Release Problem through Qualitative Reasoning", 27th IEEE International Conference on Requirements Engineering (RE'18), Banff, Aug. 2018.
- [Borgida96] Borgida A., "On the Relative Expressiveness of Description Logics and Predicate Logics", *Artificial Intelligence*, Elsevier, 1996.
- [Brachman84] Brachman R., Levesque H., "The Tractability of Subsumption for Frame-Based Knowledge Representation Languages", AAAI'84, 1984.
- [Dardenne93] Dardenne A., van Lamsweerde A., Fickas S., "Goal–directed Requirements Acquisition", *Science of Computer Programming, 20*, 1993.
- [Dung95] Dung P.M., "On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games," *Artificial Intelligence 77(2*), 321–357, 1995.
- [Elrakaiby18] Elrakaiby Y., Ferrario A., Mylopoulos J., "CaRE: A Refinement Calculus for Requirements Engineering", 27th IEEE International Conference on Requirements Engineering (RE'18), Banff, Aug. 2018.
- [Ernst10] Ernst N., Borgida A., Jureta I., Mylopoulos J., "Reasoning with Optional and Preferred Requirements", 29th International Conference on Conceptual Modeling (ER'10), Vancouver, Nov. 2010.
- [Ernst11] Ernst N., Borgida A., Jureta I., "Finding Incremental Solutions for Evolving Requirements" (submitted for publication).

References (cont'd)

- [IEEE93] IEEE Computer Society. Software Engineering Standards Committee and IEEE-SA Standards Board, IEEE recommended practice for Software Requirements Specifications, Institute of Electrical and Electronics Engineers, 1993.
- [Jackson95] Jackson M., Zave P., "Deriving Specifications from Requirements: An Example", 17th International Conference on Software Engineering (ICSE'95).
- [Jureta10] Jureta, I., Borgida, A., Ernst, N., Mylopoulos, J., "Techne: Towards a New Generation of Requirements Modeling Languages with Goals, Preferences and Inconsistency Handling", 19th Int. IEEE Conference on Requirements Engineering (RE'10), Sydney Australia, Sept. 2010.
- [Liaskos10] Liaskos, S., McIlraith, S., Sohrabi, S., Mylopoulos, J., "Integrating Preferences into Goal Models for Requirements Engineering", 19th Int. IEEE Conference on Requirements Engineering (RE'10), Sydney Australia, September 2010.
- [Nguyen15] Nguyen Chi Mai, Sebastiani R., Giorgini P., Mylopoulos J., "Multi-Objective Reasoning with Constrained Goal Models", submitted for publication.
- [Sebastiani04] Sebastiani R., Giorgini P., Mylopoulos J., "Simple and Minimum-Cost Satisfiability for Goal Models", 16th Int. Conference on Advanced Information Systems Engineering (CAiSE'04), Riga, June 2004, Springer-Verlag LNCS 2003, 20-35. Dialectics", 2016.
- [SEP16] Stanford Encyclopedia of Philosophy "Hegel's Dialectics", 2016.
- [Sebastiani15] Sebastiani R., Tomasi S., "Optimization Modulo Theories with Linear Rational Costs", *ACM Transactions on Computational Logics 16(2)*, 2015 (to appear).
- [Simon69] Simon, H., *The Sciences of the Artificial*, The MIT Press, 1969
- [vanLamsweerde09] van Lamsweerde A., "Reasoning About Alternative Requirements Options", in Borgida A., Chaudhri V., Giorgini P., Yu E., Foundations and Applications of Conceptual Modelling, Springer-Verlag LNCS 5600, June 2009.